PRIMERA REVISTA DE PROGRAMACIÓN EN CASTELLANO

PROGRAMADORES

IV, NÚMERO 42

975 Ptas.

ACTIVEX
Creación con Visual Basic 5.0

LINUX
El servicio FTP

SOFTFACTORY/2000
Solución al año 2000

DIRECTX
Curso de programación con Delphi

PROTOCOLO IP
Conoce las características de la última versión del protocolo IP

NETBIOS Y DNS
Resolución de nombres en una Intranet

AJEDREZ
Creación de tablas Hash

MÉTODO SIMPLEX
Programación Lineal

MODELADO DE OBJETOS

CONTENIDO DEL CD-ROM

- Database Designer Demo v 1.0
- Folkweb Server v1.0
- Información Eurométodo v 1
- WWW Gif Animator v 1.1
- Slackware v 3.4
- Tutorial VRML
- Y mucho más...

8 413042 303299

SÓLO PROGRAMADORES

Número 42
SÓLO PROGRAMADORES
es una publicación de
TOWER COMMUNICATIONS

Director Editor
Antonio M. Ferrer Abelló
aferrer@towercom.es
Director Adjunto
Enrique Maldonado
enriquem@towercom.es
Colaboradores
José A. Mendoza, Francisco Gayá, Enrique Díaz, Eugenio Castillo Jiménez, Ernesto Schmitz, Chema Alvarez, J.Carlos Gómez, Carlos Urbano, Constantino Sánchez, Carlos Alvaro González, Javier Toledo Fernández
Maquetación
Susana Llano
Tratamiento de Imagen
María Arce Giménez

Publicidad
Erika de la Riva (Madrid)
Tel.: (91) 661 42 11
Pepín Gallardo (Barcelona)
Tel.: (93) 213 42 29

Suscripciones
Isabel Bojo
Tel. (91) 661 42 11 Fax: (91) 661 43 86
suscrip@towercom.es

Laboratorio
Óscar Rodríguez (jefe)
Javier Amado
Servicio Técnico
Óscar Casado

Filmación
Megatipo
Impresión
Gráficas Reunidas
Distribución
SGEL
Distribución en Argentina
Capital: Huesca y Sanabria
Interior: D.G.P.

TOWER COMMUNICATIONS
Director General
Antonio M. Ferrer Abelló
Director Financiero
Francisco García Díaz de Liaño
Director de Producción
Carlos Peropadre
Directora Comercial
Carmina Ferrer
carmina@towercom.es

Redacción, Publicidad y Administración
C/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (MADRID)
Telf.: (91) 661 42 11 / Fax: (91) 661 43 86

Depósito legal: M-26827-1994
ISSN: 1134-4792
PRINTED IN SPAIN

EDITORIAL

Nuevas filosofías de distribución

Una de las últimas tendencias en las páginas Web de Internet es la colocación de miniaplicaciones, que funcionan a través del propio navegador. Esta técnica permite mejorar y personalizar sensiblemente los contenidos de Internet. Si todo funciona tal y como se preve, dentro de muy poco tiempo las aplicaciones basadas en los navegadores serán una realidad, siendo innecesario disponer de grandes sistemas de almacenamiento personal como superdiscos duros, que día a día se nos quedan pequeños.

La ventaja, de cara al usuario, es que le ofrece una forma cómoda tener su ordenador perfectamente configurado y actualizado, aunque también permitirá a los distribuidores controlar cómo funciona su software y dónde se ejecuta, algo que desagradará a los piratas informáticos y a aquellos temerosos del control informático, que sigue apuntando a la creación de un superordenador central, que empieza a recordarnos al famoso ordenador Hal de la película 2001.

La idea es muy simple y la forma de crear este mundo, al estilo de la propuesta de los NC's, no es muy compleja de llevar a cabo. Los ordenadores cliente, ejecutarán un intérprete Java y cargarán los applets del servidor cuando les haga falta. Grandes compañías, como Microsoft, ofrecen ideas alternativas, como la ejecución a través de sistemas Windows de aplicaciones, que estarán almacenadas en los servidores. Una tercera alternativa a la idea es la que ofrece Novell, quien prefiere que las aplicaciones Java se ejecuten directamente en los servidores.

Tal vez la propuesta que más fuerza tome, al menos de cara al usuario doméstico, sea la de ordenadores PC ejecutando Windows, debido a la fuerza del coloso Microsoft, con lo que no sería descabellado que viésemos dentro de unos meses ordenadores NC y terminales Windows CE compartiendo el mundo de las tecnologías de comunicación, aunque seguramente esta idea no se plasme en nuestro país hasta que las líneas de comunicación sean más baratas y ofrezcan una mejor calidad. Lo que si podemos afirmar hoy en día es que se tiende a un entorno informático en el que el usuario saldrá beneficiado, ya que los terminales propuestos no tendrán los problemas de configuración, que tienen los sistemas personales actuales, ¿o quizás si?

Tenéis a vuestra disposición la dirección de correo electrónico de la revista: solop@towercom.es, donde podéis enviar vuestras sugerencias sobre los contenidos, los temas que queráis que incluyamos y los productos que os gustaría que distribuyéramos con el CD-ROM, que todos los meses regalamos. Sois nuestros mejores colaboradores.

SÓLO PROGRAMADORES













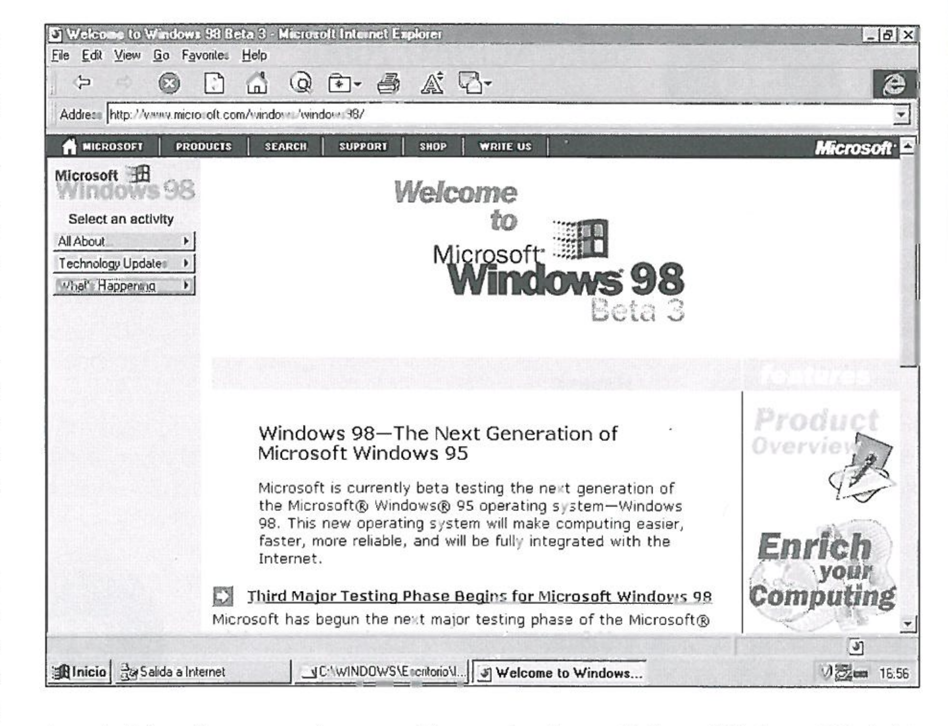




Noticias

LA PROMESA DE WINDOWS 98

La última beta de Windows 98 ha demostrado que este sistema operativo será mucho más que una nueva versión, a pesar de las grandes dificultades que está encontrando Microsoft con sus competidores en el departamento de justicia, con problemas relacionados con la inclusión de su último navegador para Internet (Inter Explorer 4.0). La solución, que ha adoptado Microsoft, es hacer opcional la inclusión de este navegador y otras características asociadas con él (Active Desktop, Canales, etc) y asegurarse que Windows 98 sea totalmente compatible con la última versión de Netscape Communicator.



La última beta de Windows 98 ha demostrado ser perfecta para que las grandes empresas empiecen a testar sus posibilidades y preparen la migración de sus aplicaciones y datos a este nuevo sistema.

Las novedades de este sistema operativo para los administradores quedan reflejadas en un conjunto de herramientas que permiten gestionar mejor la red y mantener un mayor control sobre todas las actualizaciones, tanto de programas como de controladores. Además, se ha mejorado el sistema de batch que permite crear instalaciones automatizadas (script), algo que agradecerán tanto los administradores como los ensambladores de PC's. El sistema de actualización que incluye la beta de Windows 98 permite conectarnos al Web de Microsoft y dejar que sea el propio Web, mediante controles Active X el que determine que archivos pueden ser actualizados, lo que permite que la actualización pase a ser un juego de niños. Si los administradores quieren evitar que se ande hurgando en la configuración del equipo, pueden deshabilitar. De cara a los desarrolladores, Windows 98 incluirá nuevas características como Windows Sockets 2.0, muy mejorado frente a la versión básica del código de redes de Windows.

En el lado negativo, podemos citar que los usuarios de OSR 2 y de Internet Explorer 4.0 no notarán muchas de las ventajas que incluye este sistema, ya que la mayoría de ellas ya han sido incluidas en estas versiones.

SUN Y SERMEPA-VISA ESPAÑA CREAN UN JAVA CENTER

El acuerdo permitirá a Visa España ofrecer, a sus 158 entidades financieras miembros, nuevas aplicaciones y servicios de comercio electrónico mediante el uso de tarjetas inteligentes basadas en las tecnologías Java.

El Java Center supone un acuerdo tecnológico, que permite a ambas empresas compartir sus experiencias y conocimientos en las últimas novedades tecnológicas relacionadas con Java. Se trata de un centro vivo, que irá creciendo y compartiendo sus experiencias tanto en el ámbito nacional, como en el internacional donde se podrá colaborar con otros centros, lo que producirá una sinergia de conocimientos.

Desde su nacimiento, Java es una tecnología pensada para crear una nueva generación de servicios de red que crea un software capaz de funcionar en cualquier máquina que disponga de un intérprete Java, lo que dota a Java de independencia frente al hardware. Hoy se ha dado un paso más hacia el futuro Java con la aceptación ,por parte de una gran empresa como Visa, de un futuro inminente, las JavaCard.

La JavaCard, es una tarjeta normal (como una de crédito) que puede actualizar su código en la red, utilizando las ventajas de las Java Virtual Machine.



Como muestra de la importancia, que tiene esta tecnología para SUN, se puede citar que la inversión que está realizando en la investigación de estas tecnologías supera el millón de dólares diario. Los fabricantes de chips también participan en esta tecnología, ya que serán los que creen los llamados JavaChips, que tendrán el intérprete Java.

Visa España también considera de gran interés el futuro de las JavaCard, ya que permitirá una sensible reducción en costes de mantenimiento de su TIBC, ya que hasta ahora un cambio en los programas provocaba grandes perjuicios económicos a la red de cajeros y al usuario final. La JavaCard se comporta como un NC que dispone de un procesador independiente y es la red la que trabaja y carga las aplicaciones en la tarjeta cada vez que se necesitan, lo que permite una actualización más rápida y cómoda de cara al usuario final. Así, problemas como la entrada del Euro en Julio del año 2002, se harán de una forma totalmente transparente de cara al usuario que seguiría utilizando la misma JavaCard que se le suministre el primer día. Según Visa España, se estima que sobre el año 2000 habrá más de 200 millones de tarjetas tipo JavaCard, sustituyendo a las actuales tarjetas de crédito.

Todos los usuarios y las grandes empresas tiemblan cuando se plantean los problemas de seguridad en las grandes redes, pero los planteamientos de seguridad que utilizarán las JavaCard serán los más fuertes que se han implementado hasta la fecha. Frente a las técnicas criptográficas utilizadas hoy en día, las basadas en algoritmos de clave única, se utilizarán algoritmos reversibles, que incluso pueden juntarse con otras técnicas de identificación como los escáneres de huellas. Además, la propia filosofía de las JavaCard la hacen más seguras, ya que no pueden contener virus electrónicos, ya que la tarjeta sólo contiene un intérprete de Java y seguramente una mínima información para identificar la tarjeta y el usuario en la red.

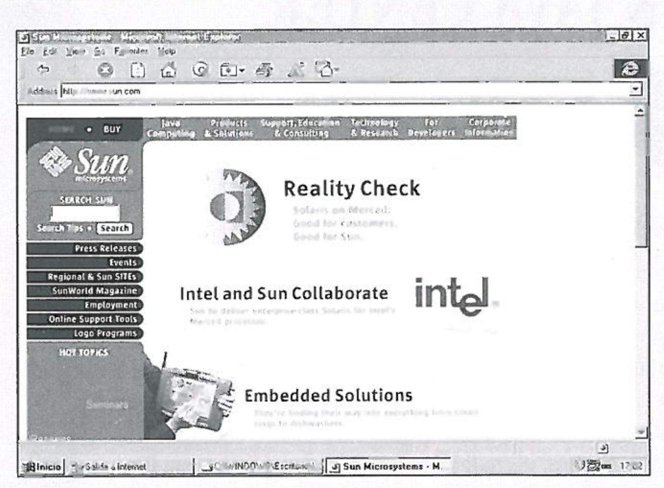
Visto todo lo anterior se puede pensar en que las JavaCard son un futuro que ya tenemos encima, el primer paso será utilizarlas como tarjetas de crédito, después se empezarán a utilizar en la sanidad pública como tarjeta sanitaria y seguramente día a día irrumpirán en las nuevas tecnologías como la televisión interactiva, los NC's, los PC, teléfonos móviles, etc.

Flash

- ✓ Silicon Graphics ha anunciado su alianza con Microsoft con el sistema operativo Windows NT. Con el acuerdo, Silicon Graphics creará PC's "visuales" basados en el interface de Windows NT, aumentando las capacidades gráficas de los sistemas y definiendo el futuro gráfico de los sistemas.
- ✓ Microsoft ha elegido ignorar la solicitud del Java Lobby (asociación libre de desarrolladores de Java), por la cual se le pedía que estudiase una compatibilidad total con la "Java core platform" (lenguaje de las virtual machines y librerías de core).
- ✓ Bay Networks ha presentado un servidor de acceso a Internet completo por 750 $, la solución ideal para conectar las pequeñas oficinas a la red de redes, ya que incluye herramientas de administración de red y funciones automáticas, como importar los servicios de directorio de Novell NetWare y Windows NT. Esta unidad consta de un módem analógico de 33,6 KB, aunque se pueden incorporar módem superiores.
- ✓ Sybase ha anunciado que su herramienta de desarrollo PowerBuilder 6.0 amplía su soporte actual con un nuevo driver que permitirá el acceso directo a la base de datos Oracle 8.
- ✓ Silicon Graphics ha batido el récord mundial de clasificación de ficheros con datos comerciales con su servidor Origin 2000, clasificando, según el benchmark de clasificación comercial MinuteSort, un terabyte (10 mil millones de registros) en dos horas y 32 minutos.
- ✓ Informática El Corte Inglés ha informatizado el registro del Ayuntamiento de Algeciras con las soluciones InvesSicres e InvesFLow que permiten el intercambio electrónico de documentos y la optimización de los flujos de trabajo respectivamente.

NUEVOS PLANTEAMIENTOS DE SUN

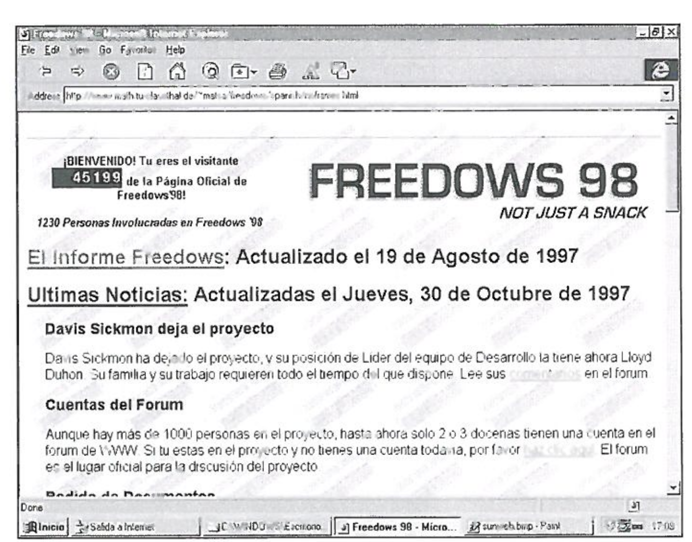
Sun Microsystems está preparando para este año planes para nuevos productos basados en las API´s de Java. Más de media docena de productos verán la luz de aquí a dos años vista, entre los que destaca un browser (navegador) especial y nuevas aplicaciones para entornos cliente/servidor. JavaSoft está preparando un plan de marketing que implicará que parte de la tecnología Java se venderá por separado. Por ejemplo, el compilador HotSpot no será incluido en la próxima versión del popular JDK.



De momento, no se ha determinado qué precio tendrá HotSpot, pero desde luego no será gratis. Otros tres productos que verán la luz gracias a JavaSoft en los próximos meses serán Activator, WebTop y HotJava Views. Activator es un nuevo control ActiveX que permite a Sun dotar de un estándar Java VM (Virtual Machine) al entorno PC . Los otros dos productos, WebTop y HotJava View, son un servidor Java y un cliente Java (un navegador). Se estima que estos productos estarán disponibles a mediados de 1998, pero los planes de JavaSoft incluyen el lanzamiento de una beta de Java run-time y del navegador HotaJava a principios de 1998.

FREEDOWS 98, EL LINUX DE WINDOWS 95/98

El proyecto Freedows sigue a buen ritmo, aunque no sin dificultades, ya que Davis Sickmon, el líder del equipo de desarrollo, ha dejado el proyecto por motivos personales. Para los que no conozcáis lo que es Freedows 98, os comentamos que se trata de un proyecto internacional que tiene como objetivo principal la creación de una versión freeware, sin costes para el usuario final, de un sistema operativo gratuito bajo la licencia pública GNU, que podrá ejecutar programas de diversos sistemas operativos simultáneamente.



Actualmente, este proyecto cuenta con más de 1000 personas, todas ellas son voluntarios y ofrecen tanto recursos humanos como recursos materiales, necesarios para completar el proyecto.

A pesar del gran número de voluntarios disponibles, todavía es necesario contar con más recursos, por lo que desde su página principal en Internet se solicita el apoyo de nuevos programadores y diseñadores.

Los interesados pueden enviar un e-mail en inglés, a freedows@bc.sympatico.ca.

Si el lector desea ampliar información de este nuevo sistema informático, puede encontrar más datos buscando "fredows" en los principales buscadores de Internet.

NUEVAS SOLUCIONES DE CISCO Y HP

Cisco Systems y Hewlett-Packard han presentado la solución SWTS. Esta solución es una combinación de aplicaciones Internet, un potente cortafuegos para los protocolos de red y una infraestructura de red de alto rendimiento. Un producto enfocado en ayudar a los clientes a aumentar su negocio en terrenos tan críticos como el comercio electrónico y los servicios financieros, en una red tan "a priori" insegura como Internet.

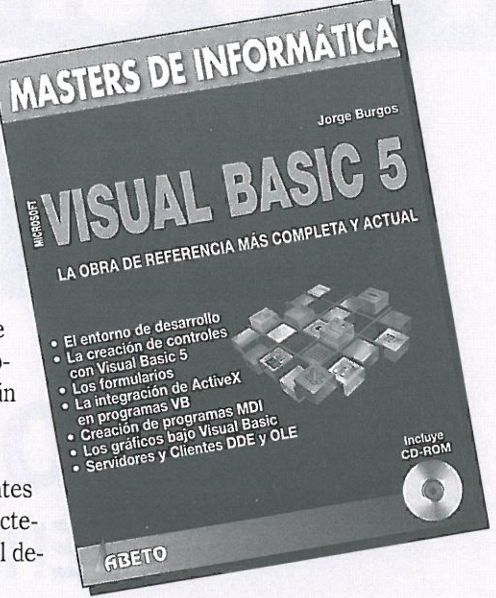
FE DE ERRATAS

El mes pasado, en el artículo "Técnica de depuración para el uso de memoria dinámica en C" se omitió el segundo autor del mismo: Alberto Postigo Gardón.

LIBROS

Microsoft Visual Basic 5

Dentro de su colección MDI (Masters de la Informática), en la que ya se han analizado Access y Unix, la editorial Abeto se centra en esta ocasión en el lenguaje de programación Visual Basic.



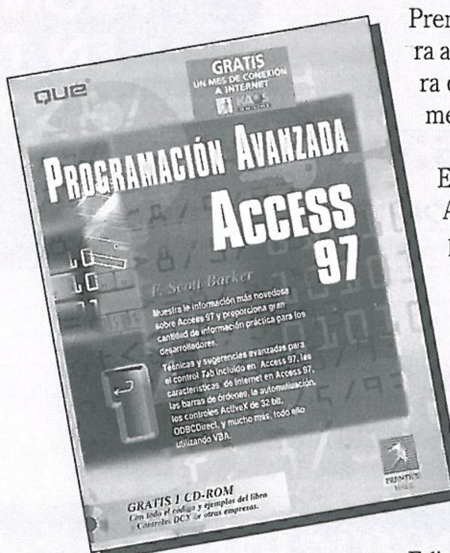
Este libro está dirigido tanto a los neófitos como a los ya iniciados en la materia. El libro sigue la línea de la serie: los temas se analizan desde su base hasta sus niveles más elevados, siempre de forma sencilla y concisa.

El producto ofrece los nuevos ejecutables compilados, y algunos de los temas que toca en profundidad son la creación de controles con este lenguaje de programación, la integración de ActiveX en programas Visual Basic, el desarrollo de aplicaciones profesionales y el uso de bases de datos, algo que agradecerán todos los programadores actuales de VBasic.

Este lenguaje de programación de Microsoft es uno de los más importantes dentro de Windows, como es sabido. Se trata de un lenguaje con algunas características, como su sencillez y su potencia, que lo hacen especialmente útil para el desarrollo de aplicaciones.

Editorial: Abeto	Autor: Jorge Burgos
Nº de páginas: 530	Idioma: castellano
PVP: 4.995 Ptas. (IVA incl.)	Nivel: medio-avanzado

Programación Avanzada Access 97



Prentice Hall continúa en su línea "avanzada", ofreciendo en esta ocasión un libro que gira alrededor de Access y sus técnicas más avanzadas. Se trata de un producto muy útil para quien quiera conocer a fondo una de las bases de datos más importantes de estos momentos.

El libro empieza por establecer los fundamentos de la Programación Avanzada en Access, para luego entrar de lleno en los secretos de esta base de datos. Entre los aspectos más destacados que son mostrados en Programación Avanzada Access 97 se encuentran el uso de objetos en esta base de datos, la automatización, la réplica, las llamadas a las APIs, los controles ActiveX, así como las características de Internet, aspecto en el que este lenguaje ha sufrido sensibles mejoras.

Intenta sentar las bases para que los desarrolladores de Access puedan adentrarse sin miedo en el campo de la programación con esta base de datos. Por lo demás, con el libro ofrecen a los usuarios un mes de conexión a Internet gratis, muy útil si queremos provar las posibilidades de comunicación que ofrece Access 97.

Editorial: Prentice Hall	Autor: F. Scott Barker
Nº de páginas: 874	Idioma: castellano
Nivel: Avanzado	Precio: Consultar



Programación Lineal. Método Simplex

José Antonio Mendoza y Francisco Gayá

La mejor manera de empezar a comprender en que consisten estas técnicas es poniendo un ejemplo muy sencillo del problema que se pretende resolver. Supongamos una empresa que produce dos tipos de productos A, y B. Con A la empresa gana digamos 540 y con B la empresa gana 125, por lo tanto el beneficio de la empresa será Ben= n*540 + m*125, es decir el número de unidades producidas de A por su beneficio mas lo mismo con B. Sin mas datos la empresa se esmerará para vender cuanto mas mejor, a mayores ventas mayores beneficios. El problema, claro está, es que la empresa, tiene unos recursos finitos, y el mercado no absorvería toda la producción, por lo tanto, no puede fabricar ilimitadamente. Nace por tanto la necesidad de asignar dicha capacidad de producción, entre los distintos productos que la empresa produce. Dicha asignación viene modulada por una serie de restricciones, las cuales consisten en una serie de inecuaciones, como por ejemplo, en fabricar A, la empresa tarda tres horas y en fabricar B la empresa tarda una y media, y el número de horas totales de que dispone la empresa en el periodo es de 900. Otra Restricción puede ser el volumen de almacenaje de productos terminados de la empresa, y el volumen de cada uno de ellos.

Esto se entiende muy fácilmente en una factoría de coches, cuando se llena el aparcamiento de los coches recién fabricados, la fabrica tiene que parar necesariamente, entrando en modo pánico nuestro jefe de factoría por no haber previsto tal circunstancia. Retomando nuestro ejemplo Si el volumen de nuestra empresa es de 1500 y A ocupa diez y B ocupa dos ya tendríamos una nueva restricción que tener en cuenta. Resumiendo, tendríamos la función a maximizar, el beneficio :

Ben=n*540+m*125

y dos restricciones :

n*3+m*1.5 <= 900
n*10+m*2 <= 1500

lógicamente :

n,m>0

Ya que la empresa no va a vender productos con signo negativo. Por tanto, ¿cúal es el número de productos de A y B que hacen máximo el beneficio de la empresa? Ese será nuestro objetivo, obtener la cantidad producida de cada uno de los productos. Para ello, disponemos de dos métodos fundamentalmente, uno gráfico para solucionar problemas sencillos, como el anterior, de solo dos incógnitas y otro, el Simplex, para resolver problemas de muchas mas variables.

El método gráfico consistiría en dibujar dos rectas y ver el punto de intersección de ambas. Esa sería la combinación de productos a producir que nos

Ampliamente utilizada por ingenieros y economistas, la programación lineal trata de resolver el problema de mejor asignación, el cual, consiste en dados unos recursos limitados asignarlos de tal manera que el objetivo pretendido se alcance con el mayor rendimiento posible.

Listado 1

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

int main ( ) {
        unsigned n,e,*x;
    float *c,**a,*h,*v;
    unsigned i,j,bi,bj;

    /* Toma de datos */
    printf ("SIMPLEX\n\n");
    printf ("Num. var.: ");
    scanf ("%u",&n);
    printf ("Num. inec.: ");
    scanf ("%u",&e);
    c = calloc ( n+1, sizeof( float));
    a = calloc ( e+1, sizeof( float*));
    x = calloc ( e+1, sizeof( unsigned));
    h = calloc ( n+1, sizeof( float));
    v = calloc ( e+1, sizeof( float));
    for ( i=0;i<n;i++) {
        printf ("Coef. x(%u) en la ec. del Maximo:",i+1);
        scanf ("%f",&c[i]);
    }
    for ( j=0;j<e;j++) {
        a[j] = calloc ( n+1, sizeof(float));
        for ( i=0;i<n;i++) {
            printf ("Coef. x(%u) en la %u ec.:",i+1,j+1);
            scanf ("%f",&a[j][i]);
        }
        printf ("Term.Indep. de la %u ec.:",j+1);
        scanf ("%f",&a[j][n]);
        printf ("Subindice de variable aux. de la inecuacion h(i)");
        scanf ("%u",&x[j]);
        x[j]—;
    }

    /* Preparacion de la matriz de datos */
    for ( i=0; i<=n;i++ ) {
        a[e][i]=-c[i];
        for ( j=0;j<e;j++) {
            a[e][i]+=a[j][i]*c[x[j]];
        }
    }

    for (i=0; i< n; i++ ) {
```

reportaría un beneficio máximo. Basta con verlo en la figura 1.

En ella apreciamos las dos regiones del plano limitadas por las dos inecuaciones, con n y m mayores que cero. El punto de intersección de las rectas, que obtenemos de convertir las dos inecuaciones en igualdad será el número de productos A y B que necesitamos producir para obtener el mayor beneficio posible, limitado este a nuestras posibilidades.

Método Simplex

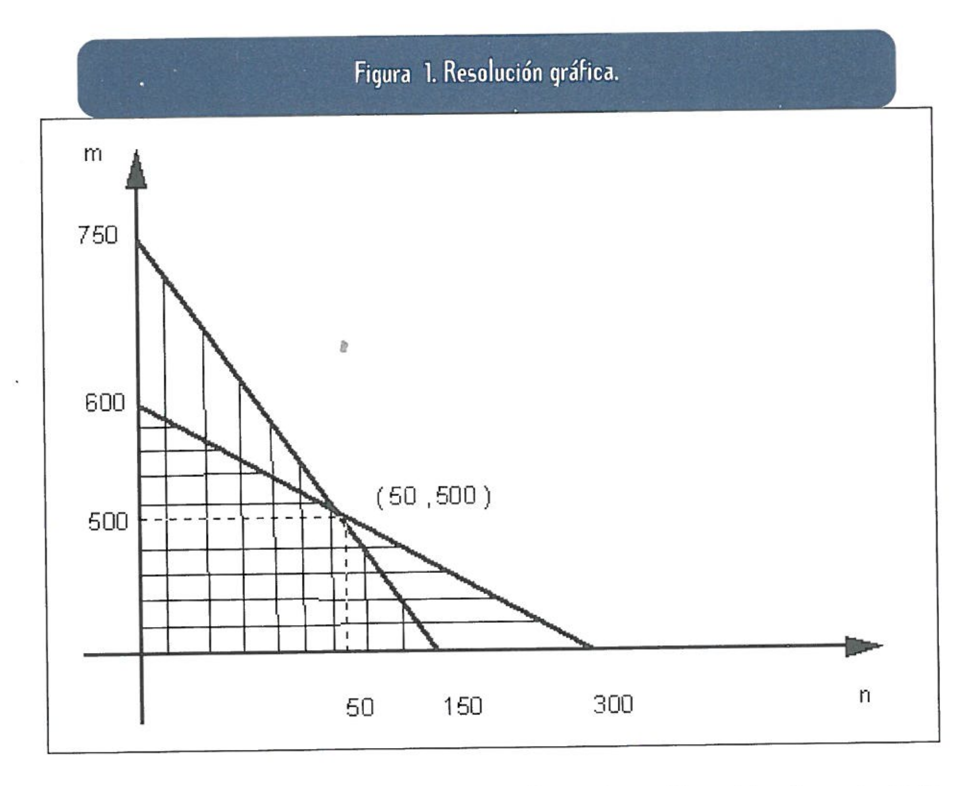
Cuando tenemos un número de variables superior a dos, el método gráfico se nos queda escaso, teniendo que recurrir a técnicas mas potentes. Disponemos entre ellas del método de resolución Simplex.

El problema a resolver, en el ejemplo anterior, fue la cantidad de productos a fabricar de entre todas las posibles que nos diera un mayor beneficio. Generalizando la problemática el método nos resolverá como maximizar una función de n variables que cumplan una serie de limitaciones.

Gracias a la programación lineal se obtiene la maximización de una función lineal

Es el momento de algunas definiciones, las imprescindibles, para sin profundizar mucho, tener los conceptos claros. En primer lugar a $F(x_1 ,x_2 ,x_3 ,\ldots\ldots, x_n)$ la llamaremos función objetivo, y no tiene por que ser un beneficio, puede representar cualquier magnitud. F no puede ser cualquier cosa, es mas, está restringida a tener forma lineal, es decir, una combinación lineal de las variables :

Figura 1. Resolución gráfica.



$F = A_1x_1 + A_2x_2 + A_3x_3 + ... + A_nx_n$

Donde los A_i son los coeficientes y las x_i son las cantidades de cada una de las magnitudes involucradas. Como hemos dicho anteriormente las cantidades son positivas, si al resolver nos sale alguna negativa es que nos hemos equivocado. Vista la función objetivo veamos unos pocos conceptos de las restricciones. Cada restricción puede ser de igualdad, de mayor igual o de menor igual. Cuando solo tengamos "=" tenemos la forma standard, cuando tengamos "<=" la llamaremos forma canónica y cuando se den las dos forma mixta. Las condiciones de igualdad son equivalentes a una de ">=" y otra de "<=", y estas son equivalentes la una a la otra multiplicando por menos los dos términos, es decir si 4>=2, -4<=-2, así todas las condiciones se pueden llevar a forma canónica.

En este momento tenemos planteado el problema y nos falta la manera de resolverlo. George B. Dantzing en 1947 desarrolló el método Simplex, que de forma iterativa llega a la solución que maximiza la función objetivo. Es el momento de explicar el método y posteriormente explicar el funcionamiento de los programas, el de C, y el de Basic.

Lo primero que se hace, y que vamos a considerar dado, es la obtención de la función objetivo, que no suele ser tarea fácil, pues para llegar a ella hace falta obtener los Ai del mundo real. Llegar a la conclusión de que de tal producto, ganamos x dinero no suele ser trivial, la complejidad de la teoría de costes nos da idea de lo complicado del objetivo. Pero para nosotros la función objetivo es dato y las restricciones también. Para llevar estas ecuaciones a algo con lo que poder operar, nos vamos a servir de las llamadas variables de holgura h_i. A cada una de las restricciones le añadiremos una de estas variables, pudiendo pasar de la inecuación a una ecuación normal. Recordemos que las restricciones con igual son transformadas en dos inecuaciones. Llegaríamos al siguiente sistema :

$a_{11}x_1 + a_{12}x_2 + a_{13}x3 + ...+ a_{1n}x_n + h_1 = b_1$

$a_{21}x1 + a_{22}x_2 + a_{23}x_3 + ...+ a_{2n}x_n + h_2 = b_2$

....

$a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + ...+ a_{mn}x_n + h_m = b_m$

Si la función objetivo es :

$Z = A_1x_1 + A_2x_2 + A_3x_3 + ... + A_nx_n$

Formaríamos la siguiente tabla :

b_1 $a_{11}\ a_{12}\ a_{13}\ ....a_{1n}$ 1 0 0.....0

b_2 $a_{21}\ a_{22}\ a_{23}\ ....a_{2n}$ 0 1 0.....0

b_3 $a_{31}\ a_{32}\ a_{33}\ ....a_{3n}$ 0 0 1.....0

....

b_m $a_{m1}\ a_{m2}\ a_{m3}\ ....a_{mn}$ 0 0 0.....1

$-A_1\ -A_2\ -A_3.........-A_n$ 0 0 0.....0 A esta fila se la llama regleta de control.

En resumen, las "bi" las ponemos las primeras por la izquierda, a continuación ponemos los $a_{i,j}$ y seguido de estos el uno y los ceros, es decir, un uno por el coefi-

Figura 2. Toma de datos.

```
C:\BAS\ECO\simplex>simplex
SIMPLEX

Num. var.: 2
Num. inec.: 2
Coef. x(1) en la ec. del Maximo:540
Coef. x(2) en la ec. del Maximo:125
Coef. x(1) en la 1 ec.:3
Coef. x(2) en la 1 ec.:1.5
Term.Indep. de la 1 ec.:900
Subindice de variable aux. de la inecuacion h(i)3
Coef. x(1) en la 2 ec.:10
Coef. x(2) en la 2 ec.:2
Term.Indep. de la 2 ec.:1500
Subindice de variable aux. de la inecuacion h(i)4
```

Listado 1 (Continuación)

```
printf ("c(%u)= %7.2f\n", i+1,c[i]);
}

/* Bucle de resolucion */
printf ("\nSolucionando...\n");
while ( 1 ) {
	/* Obtencion de la columna del elemento pivote */
	bi=1;
	for ( i=0;i<n;i++) {
		if ( a[e][i]< a[e][bi] ) bi=i;
	}
	if ( a[e][bi]>=0 ) break;
	/* Obtencion de la fila del elemento pivote */
	bj=1;
	for ( j=0;j<e;j++) {
		if ( a[j][n]*a[bj][bi] < a[bj][n]*a[j][bi] ) bj=j;
	}
	printf ("+");
	for ( i=0;i<n+2;i++)  printf ("———————");
	printf ("+\n");
	for ( j=0; j< e; j++) {
		printf ("| x%u | %7.2f | ",x[j]+1,a[j][n] );
		for ( i=0; i<n; i++      ) {
			if ((i==bi)&&(j==bj)) {
				printf ("%7.2f*",a[j][i] );
			}
			else {
				printf ("%7.2f ",a[j][i] );
			}
		}
		printf("|\n");
	}
	printf ("+");
	for ( i=0;i<n+2;i++)  printf ("———————");
	printf ("+\n");
	printf ("    | %7.2f | ",a[e][n] );
	for ( i=0; i<n; i++      ) {
		printf ("%7.2f ",a[e][i] );
	}
	printf("|\n\n");

	/* Guardar los datos de la fila y columna (cruz) del elemento pivote */
	for (i=0;i<=n;i++ ) {
		h[i]=a[bj][i];
	}
	for ( j=0;j<=e;j++ ) {
		v[j]=a[j][bi];
```

ciente de la h_i correspondiente y los ceros por la no presencia de las demás h_i. Los A_i, los hemos puesto cambiados de signo y seguidos de ceros, pues en la función objetivo no aparece ningún h_i.

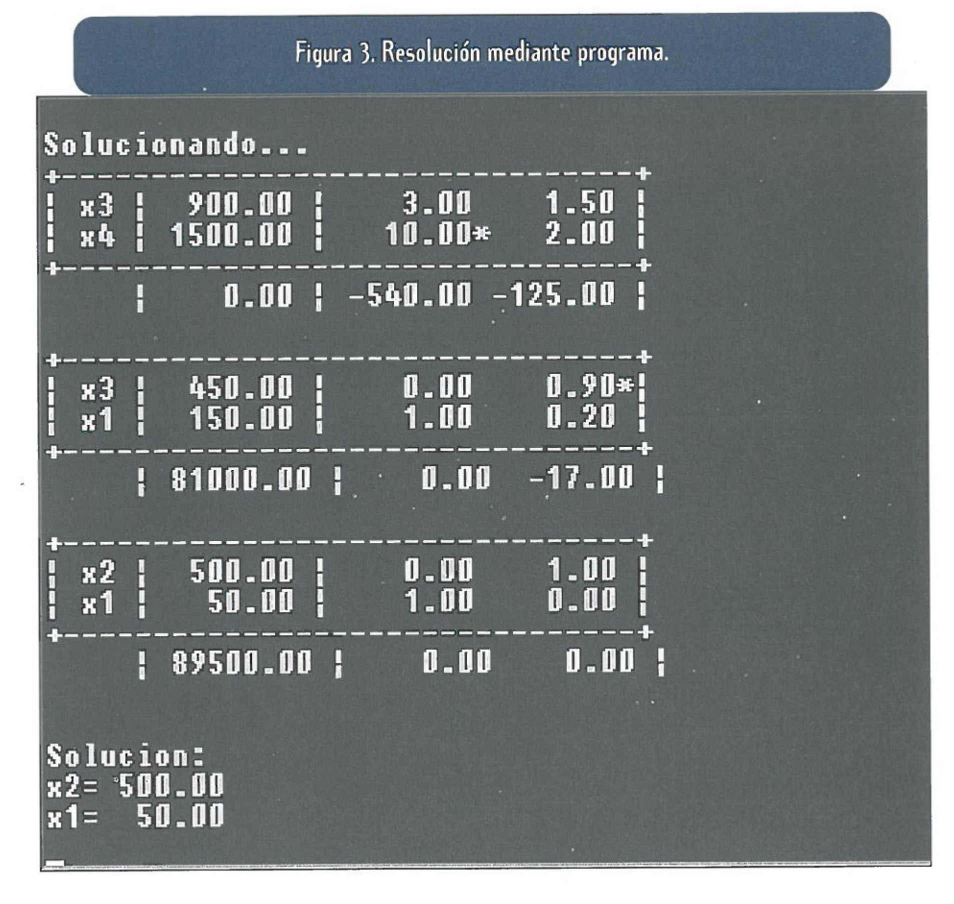
Una vez puestos los datos de esta manera ya podemos empezar a operar. Lo primero que calcularemos será determinar el elemento pivote, pieza clave del algoritmo. Para ello buscamos el menor de los números de la regleta de control, recordemos que el menor de dos números es aquel que está mas próximo al cero si son positivos, pero si son negativos será menor el que esté mas alejado de el, 1<2 pero -1>-2.

Encontrado el menor de los A_i, será en esa columna donde se encuentre el elemento pivote, para hallarlo tenemos que dividir cada b_i por su correspondiente $a_{i,j}$. El elemento pivote será el $a_{i,j}$ del menor de los cocientes $b_i/a_{i,j}$. Obtenido pues el elemento pivote nos quedan dos pasos, el primero dividir por el pivote toda la fila a la que pertenece incluido el mismo, y después hacer ceros todos los elementos de su columna, actualizando los valores de los elementos de las demás columnas. Esto lo haríamos hasta conseguir que en la regleta de control no queden números negativos y la solución sería la columna de b'_i, es decir los valores que quedan después de hacer todas las operaciones necesarias.

Método desarrollado en 1947 por Geoge Dantzing

Para entender mejor el método veamos el ejemplo anterior, que habíamos solucionado de manera gráfica, pero ahora con el Simplex. La función objetivo era $Z= 540x_1+125x_2$ y las restricciones, $3x_1+1.5x_2<= 900$ y $10x_1+2x_2 <= 1500$, empecemos a resolver formando la tabla, y añadiendo dos variables de holgura :

Figura 3. Resolución mediante programa.



900	3	1.5	1	0
1500	10	2	0	1
	540	-125	0	0

450	0	0.9	1	-0.3
150	1	0.2	0	0.1
	0	-17	0	54

Primero veamos cual es la columna donde se encuentra el elemento pivote, el número mas pequeño es el -540, por lo que el elemento pivote será el 3 o el 10, para ver cual de ellos es calculemos los cocientes : 900/3=300 y 1500/10=150. 150<300 luego el elemento pivote será el 10. Segundo, encontrado el pivote dividimos por este toda su fila :

900	3	1.5	1	0
150	1	2/10	0	1/10
	-540	-125	0	0

Tercero, tenemos ahora que hacer cero toda su columna excepto el mismo, para ello la fila del elemento pivote la multiplicamos por tres y se la restamos a la primera, haciendo lo mismo con la regleta de control, pero esta vez multiplicamos por 540 y sumamos nos quedaría :

Vemos en la regleta de control un número negativo, por lo tanto el método no ha terminado, hacemos lo mismo que antes pero ahora con la columna del menos diecisiete. 450/.9=500 y 150/.2=750 el 0.9 nuevo elemento pivote nos queda tras normalizar:

500	0	1	1/0.9	-1/3
150	1	.2	0	0.1
	0	-17	0	54

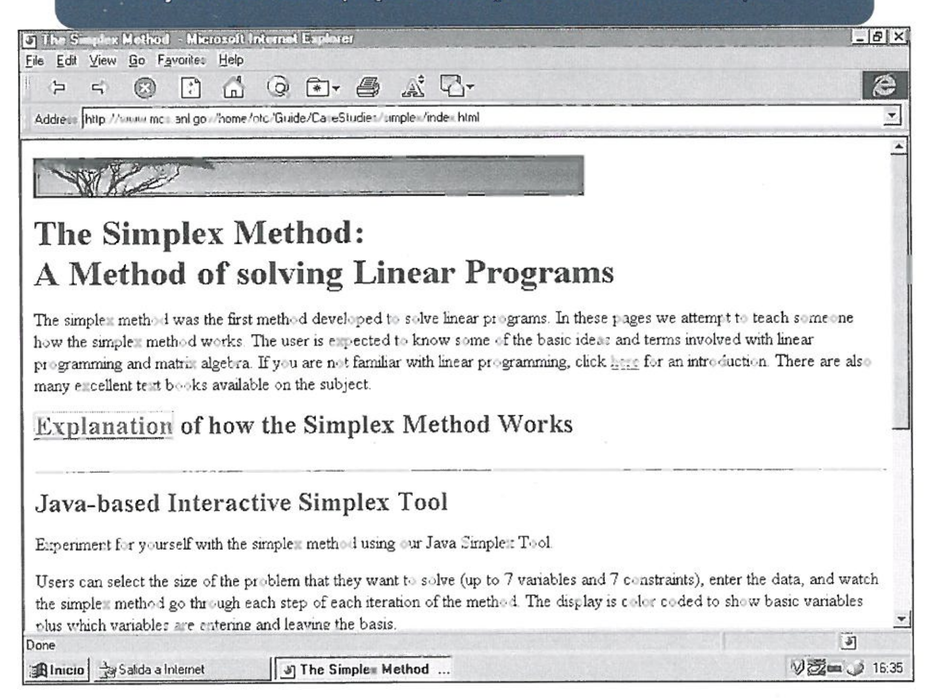
Y tras hacer cero la regleta de control :

50	0	1	1/0.9	-1/3
50	1	0	2/9	0.03
	0	0	17/0.9	48.3

La solución por tanto será 500 y 50.

En el CD-ROM vienen las fuentes de los programas en C y Basic que resuelven este método. Las fuentes de C las mostramos a continuación. Veamos un poco su funcionamiento. En primer lugar el programa pide el número de variables, en nuestro caso dos, después el número de inecuaciones, dos, acto seguido los coeficientes de la función objetivo y de las inecuaciones. Queda reseñar la introducción de h(i), el subín-

Figura 4. URL con un programa Java, que resuelve el método Simplex.



Listado 1 (Continuación)

```
            }
            /* Transformacion de la matriz */
            for (i=0;i<=n;i++ ) {
                    for ( j=0;j<=e;j++ ){
                            a[j][i] -= h[i]*v[j]/h[bi];
                    }
            }
            for (i=0;i<=n;i++ ) {
                    a[bj][i]=h[i]/v[bj];
            }
            x[bj]= bi;
    }

    /* Presentacion de solucion */
    printf ("+");
    for ( i=0;i<n+2;i++)  printf ("————");
    printf ("+\n");
    for ( j=0; j< e; j++) {
            printf ("| x%u | %7.2f | ",x[j]+1,a[j][n] );
            for ( i=0; i<n; i++      ) {
                    printf ("%7.2f ",a[j][i] );
            }
            printf("|\n");
    }
    printf ("+");
    for ( i=0;i<n+2;i++)  printf ("————");
    printf ("+\n");
    printf ("     | %7.2f | ",a[e][n] );
    for ( i=0; i<n; i++       ) {
            printf ("%7.2f ",a[e][i] );
    }
    printf("|\n\n");
    printf ("\nSolucion:\n");
    for ( j=0; j<e; j++) {
            printf ("x%u=%7.2f\n",x[j]+1,a[j][n]);
    }

    /* Liberar espacio de variables */
    for ( j=0; j<=e; j++) free (a[j]);
    free (c);
    free (a);
    free (x);
    free (h);
    free (v);
    return 0;
}
```

dice de la variable de holgura en la primera inecuación tres y en la siguiente cuatro y así sucesivamente si hubiera mas inecuaciones. La primera variable de la función objetivo tiene subíndice uno la segunda dos y la primera variable de holgura tres la segunda cuatro. Ese sería el subíndice uno mas que el número de variables.

Cuando el número de variables es superior a dos el método gráfico no sirve

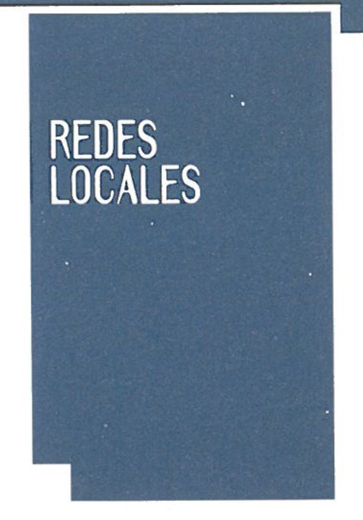
En cuanto a la versión de Basic solo varía en que el número de variables que pide el programa es ahora el total es decir las de la función objetivo mas las de holgura. Siendo el subíndice el mismo que en el caso anterior.

Conclusión

Gracias a la programación lineal los directivos de las empresas, pueden optar por las estrategias de producción y asignación de recursos, políticas de costes y beneficios y encontrar la manera óptima de funcionamiento de la empresa. Así mismo los ingenieros en sus cálculos pueden asignar los recursos, siempre escasos, de la manera mas eficiente, para ello el método Simplex es la herramienta que ayuda a la obtención de la solución. Como hemos dicho al principio, lo verdaderamente difícil es la obtención de la función objetivo y las restricciones a partir de la realidad.

Bibliografía

- " Dirección de operaciones : Aspectos estratégicos en la producción" Autor : José A. Dominguez Manchueca.



Resolución de nombres de Internet

Enrique Díaz Trobo

enrique@nova.es

El mes anterior dejamos más o menos configurado el protocolo TCP/IP para nuestra intranet. Tal como lo dejamos sólo podremos acceder a los ordenadores de la red TCP/IP especificando su dirección IP, por ejemplo 195.180.123.1. Algo muy duro, muy inhumano y muy difícil de recordar. Para evitar esto, lo que haremos es vincular esas direcciones IP a nombres que nos resulten más fáciles de recordar.

Así pues, en primer lugar relacionaremos las direcciones IP con nombres NetBIOS propios de nuestra red, y más tarde entraremos con los nombres de Internet. Los nombres NetBIOS son planos, es decir, constan de una única parte y no de varias separadas por puntos, a diferencia de los de Internet, que utilizan una convención de nombres denominada *Sistema de nombres de dominio* (DNS). Los nombres de dominio constan de dos partes: un *nombre de host* y un *nombre de dominio*, que conforman el llamado *nombre completo de dominio* (FQDN). En nuestra intranet, el mismo ordenador tendrá los dos nombres. Por ejemplo un mismo ordenador será marivi y también marivi.diazmarcos.org. Para llevarlo a cabo instalaremos dos servidores más: un servidor WINS, para resolver los nombres NetBIOS y otro DNS para los nombres de Internet. Y no sólo eso; también haremos que ambos servidores trabajen juntos, activando búsquedas inversas. Si le preguntamos a un servidor por un nombre que no reconoce, preguntará al otro servidor para ver si puede resolverlo.

Nombres NetBIOS

El sistema plano de nombres de NetBIOS viene a ser como las zapatillas de andar por casa; son muy cómodas y confortables, pero cuando ponemos un pie fuera, no ocurre lo mismo. Del mismo modo, los nombres de NetBIOS pueden ser muy cómodos en una pequeña red local pero en cuanto la red crece un poquito presentan serios problemas para su administración:

- El conjunto de nombres que podemos establecer es mucho más limitado y corremos el riesgo de repetirlos.
- Para administrarlos necesitaremos de una autoridad central que asegure que cada nombre de equipo es único en la red y que se ocupe de organizar el tema de las altas y bajas de los equipos en la red. La carga administrativa que esto produce es tanto mayor cuanto más grande sea la red.

Nombres DNS

El sistema de nombres de dominio esta jerarquizado, que no centralizado. Una manera gráfica de entenderlo es imagi-

En este artículo veremos cómo hacer para que nuestra intranet resuelva tanto nombres NetBIOS como DNS. De este modo quedará todo preparado para recibir el servidor de Web.

Tabla 1. Ejemplo de HOSTS (para DNS)

195.180.123.1	dafne.diazmarcos.org
195.180.123.1	dafne
195.180.123.15	marivi.diazmarcos.org
195.180.123.15	marivi
195.180.123.12	bbs.diazmarcos.org
195.180.123.12	bbs

narlo como un árbol con la copa hacia abajo. Existe una autoridad central administra la parte superior de este árbol invertido, pero las ramas se administran a sí mismas y son ellas las que se ocupan de que los nombres sean únicos en sus respectivas redes y subredes.

Hasta no hace mucho, la forma de implementar la relación entre nombres de equipo y su dirección IP se hacía por medio de archivos de configuración. En el caso de DNS el archivo se llama HOSTS, y en el caso de NetBIOS LMHOSTS. Estos dos archivos no son otra cosa que archivos de texto ASCII en los que se representa una tabla de nombres y direcciones IP (en las tablas 1 y 2 tenéis un ejemplo de HOSTS y LMHOSTS, respectivamente).

Para que la cosa funcione es necesario que cada ordenador de la red tenga una copia exactamente igual de estos archivos. Aunque efectivo, y aún hoy perfectamente recomendable para redes pequeñas, este sistema ocasiona múltiples problemas a la hora de administrarlo: mantener todos los archivos sincronizados y actualizados supone una importante carga administrativa, amén de que es difícil garantizar que cada equipo tiene un nombre único en la red Por otra parte, los efectos de tener dos o más ordenadores con el mismo nombre son imprevisibles.

Tabla 2. Ejemplo de LMHOSTS (Para NetBT)

195.180.123.1	dafne
195.180.123.15	marivi
195.180.123.12	bbs

La resolución de nombres y el rendimiento de la red

A la hora de relacionar nombres de equipo y direcciones, existen dos problemas que deberemos solucionar: el registro, que es el proceso mediante el cual asignamos un nombre único a un ordenador o nodo, y la resolución, que es el proceso para determinar la dirección correspondiente a un nombre de equipo.

En Windows NT esto se soluciona mediante un servicio a nivel de sesión (en cuanto a la pila OSI) denominado NetBT. La implantación de éste servicio la realizaremos con el *Servicio de nombres de Internet de Windows* (WINS).

El sistema de nombres de dominio está jerarquizado, que no centralizado

En el número anterior vimos cómo el servidor DHCP asignaba a los clientes un tipo de nodo:

- nodo b, que utiliza la difusión de mensajes para resolver nombres
- nodo p, que utiliza comunicaciones punto a punto con un servidor de nombres para la resolución
- nodo m, que utiliza primero el nodo b y si lo consigue, intentará el nodo p para resolver nombres
- nodo h, que utiliza primero el nodo p para las consultas de nombres y después el nodo b si el servicio de nombres no está disponible o si el nombre no está registrado en la base de datos

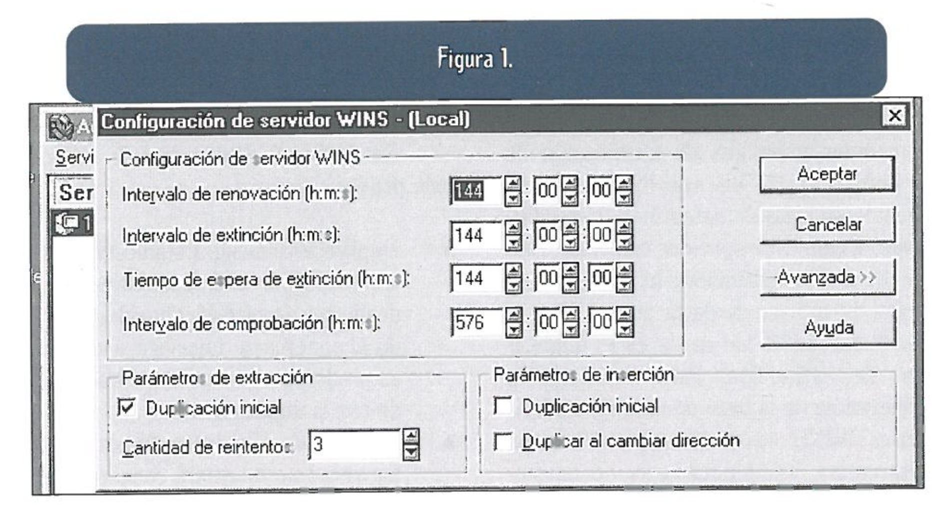
Veamos como funcionan los tipos de nodo:

- El *nodo b* utiliza mensajes de difusión para el registro y la resolución de nombres. Esto quiere decir que, para comunicarse con otro nodo, difundirá a todos los sistemas un mensaje indicando que está buscando a un equipo, y esperará el tiempo especificado para que el equipo con el cual quiere comunicarse le responda. Este modo tiene el inconveniente de que se carga la red con difusiones. Un problema añadido es que los encaminadores no reenvían difusiones, por lo que los equipos que se encuentran en los extremos opuestos de un encaminador nunca responderán a estas solicitudes. Posteriormente se ha creado el llamado *nodo b modificado.* En este entorno, si falla el nodo b se utilizará el archivo LMHOSTS —en el caso de NetBIOS— para cruzar el encaminador.

- El modo *nodo p* intenta solucionar los problemas que el nodo b no resuelve. En un entorno de nodo p, se eliminan las difusiones, ya que todos los nodos se registran en el servidor WINS, que es el que conoce los nombres y las direcciones de los equipos, asegurando al mismo tiempo que no haya nombres duplicados en la red. En este entorno, cuando un nodo desea comunicarse con otro, sencillamente le pregunta la dirección al servidor WINS. Una vez obtenida la dirección, conecta directamente con él sin utilizar difusiones. De esta manera se solucionan los problemas del nodo b: no se utilizan difusiones que carguen la red y los equipos pueden estar en extremos opuestos a los encaminadores.

Una de las leyes de Murphy dice que si observamos cuatro posibles problemas y ponemos los medios para evitarlos, se generará un quinto problema no previsto. Así que con la solución a los problemas anteriores hemos generado algunos nuevos derivados de una dependencia excesiva del servidor WINS, ya que todos los equipos deben conocer la

Figura 1.



dirección del servidor WINS y, si por casualidades de la vida, el servidor WINS no está operativo, los equipos que utilicen este servidor para resolver las direcciones no podrán llegar a los demás sistemas de la red.

- De esta manera llegamos al *nodo m*, que pretende solucionar los problemas que se plantean con los nodos b y p. En un entorno de nodo m, un equipo intenta el registro y la resolución primero con el nodo b. Si falla, cambia al nodo p. Con esto se consigue que el nodo m pueda franquear los encaminadores y que en caso de fallo del servidor WINS al menos funcionarán normalmente los equipos que están en el mismo extremo de un encaminador.
- Esto nos devuelve, en parte, a uno de los problemas del nodo b: las difusiones. Para mejorarlo tenemos el *nodo h*, que viene a ser un nodo m al revés; esto es, intenta primero como nodo p y si el servidor WINS no funciona, prueba como nodo b utilizando las difusiones. Si fallara como nodo p, el nodo h intentará periódicamente establecer contacto con el servidor WINS. Si lo consigue, cambiará automáticamente a nodo p, de modo que dejará de utilizar las difusiones. Además, en un nodo h podemos especificar un archivo LMHOSTS si falla también la resolución de nombres por difusión.

Instalación de WINS

Para utilizar WINS en una red, primero debemos instalar el servidor WINS en cualquiera de los servidores NT. Para ello, ejecutaremos el icono Red del *Panel de control*, marcaremos *Servicios* y después *Agregar*. Seleccionamos *Servidor de nombres de Internet para Windows* y aceptamos. Tras la copia, reiniciaremos el equipo y se agregará la opción *Administrador de WINS* en *Herramientas administrativas (Común)*.

Desgraciadamente, WINS es un protocolo exclusivo de Microsoft, y los clientes ajenos a su red no pueden utilizarlo, de modo que habremos de distinguir tres tipos ordenadores en las redes que lo utilicen:

- Servidor WINS. Podemos tener varios servidores WINS en una red, y de hecho es recomendable tener al menos dos, por si uno de ellos falla. Cada uno de los servidores de WINS puede duplicar las bases de datos con otros servidores para mejorar el rendimiento de la red.
- Clientes WINS. Sólo los equipos con Windows NT, Windows 95 o Windows 3.11 para trabajo en grupo pueden interactuar directamente con los servidores WINS.
- Clientes no WINS. Los clientes configurados como nodo b pueden interactuar con clientes WINS que se hallen en su mismo segmento de red. Los clientes WINS obtienen información del servidor WINS y proporcionan esta información a los clientes no WINS. De esta manera, los clientes WINS funcionan como *proxies* de clientes no WINS. Como es natural, deberá existir al menos un cliente WINS en el segmento de red.

Tabla 3.

Dominio	Actividad
com	Comercial
edu	Educativa
gov	Gubernamental
org	Organizaciones no comerciales
net	Organizaciones de conectividad en red
mil	Militar
int	Internacional

Otra de las cosas a tener en cuenta es asignar una dirección IP fija a cada servidor WINS. Aunque no es necesario que sean clientes DHCP, si deberemos reservarles la dirección si usamos alguno en la red.

Duplicación de servidores WINS

Si optamos por poner más de un servidor WINS en la red, los servidores se comunicarán entre sí para duplicar sus bases de datos. De este modo un nombre registrado en cualquiera de ellos se duplicará en el resto de los servidores, reduciendo nuestra dependencia de una sola máquina.

Un servidor de WINS puede actuar como duplicador de inserción o como

duplicador de extracción, respecto a otro servidor WINS.

El duplicador de inserción es aquel que envía mensajes de notificación de actualización a su servidor asociado cuando su base de datos ha sido modificada. Cuando el servidor asociado responda a la notificación, le enviará una copia de su base de datos actual. El servidor que envía los datos es el duplicador de extracción. Para mantener la coherencia de la base de datos los servidores WINS se convierten, de forma alternativa, en duplicadores de extracción e inserción, los unos con respecto a los otros.

En caso de necesidad, también podemos activar la duplicación manualmente, o cuando se alcancen los umbrales definidos por el administrador.

La configuración del servidor

Para configurar el servidor abriremos el menú *Servidor* y pincharemos sobre *Configuración*. Se nos abrirá una pantalla como en la figura 1 en la que vemos los siguientes apartados:

- Intervalo de renovación. Aquí indicaremos la frecuencia con la que un cliente vuelve a registrar su nombre. Si pasado ese tiempo no se registra el nombre la entrada se marca como *liberada*.
- Intervalo de extinción. Establecemos cuánto tiempo pasará desde que la entrada se marcó como *liberada* hasta que se marque como *extinta*.
- Tiempo de espera de extinción. Establecemos el intervalo entre cuando una entrada se marca como *extinta* hasta que se borra definitivamente de la base de datos.
- Intervalo de comprobación. Indicamos el intervalo después del cual el servidor WINS tiene que comprobar que los nombres viejos que no son de su propiedad están aún activos.

En el bloque *Parámetros de extracción* podemos indicar:

- Duplicación inicial. Establecemos si este servidor extrae duplicaciones desde sus asociados conocidos cuando el sistema se inicialice o cuando se produzca algún cambio relacionado con la duplicación.
- Cantidad de reintentos. Indica cuantas veces se intentará conectar con un asociado, en caso de que se produzca algún fallo.

Para *Parámetros de inserción* especificaremos:

- Duplicación Inicial. El servidor informará a los duplicadores de extracción del estado de su base de datos cuando el sistema se inicialice.
- Duplicar al cambiar de dirección. El servidor informará a los duplicadores de extracción cuando cambie una dirección en el registro de asignación.

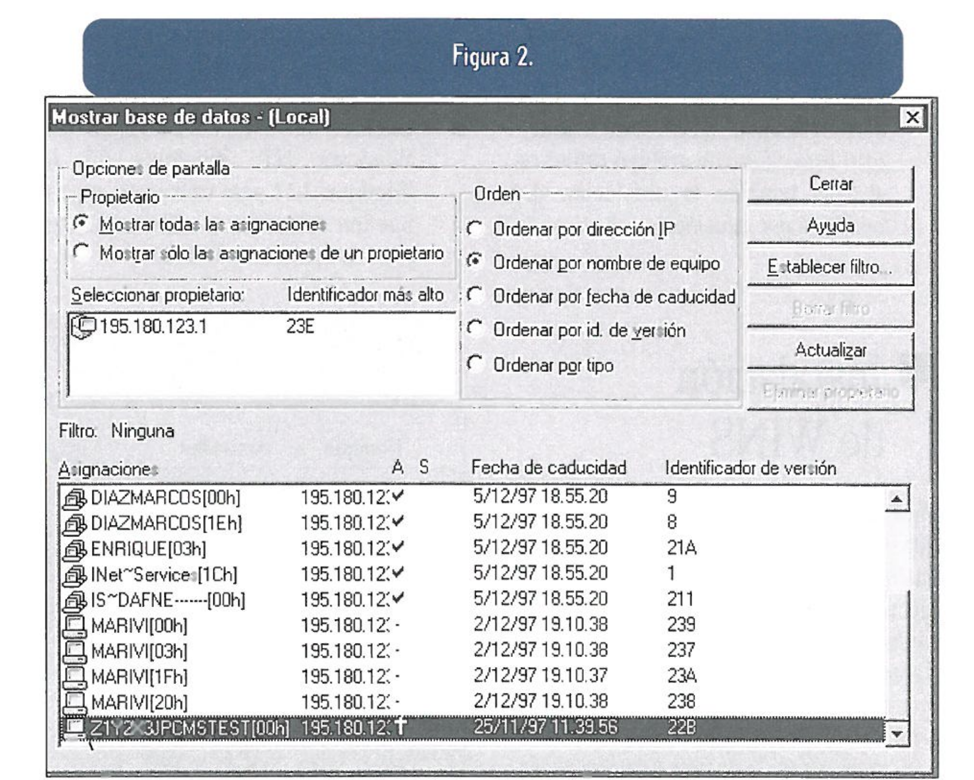
La base de datos

Para ver la base de datos de WINS, abriremos el menú *Asignaciones* y elegiremos la opción *Mostrar base de datos*.

Bajo el epígrafe *Asignaciones* encontraremos la lista de los registros de la base de datos. El icono de la izquierda (figura 2) nos indica si se trata de una asignación exclusiva (el icono representa un único ordenador) o si se trata de grupos, nombres de dominio, multitarjeta o grupo de Internet (el icono representa varios ordenadores). Asimismo, podremos ver el nombre del equipo NetBIOS y la dirección IP asignada al equipo. La Columna *A* nos indica que la asignación está activa, y la columna *S* nos indica que se trata de una asignación estática. La fecha de caducidad nos muestra la fecha en la que caducará el registro.

El identificador de versión, nos muestra un número hexadecimal único asignado por el servidor WINS durante el registro de nombres, que será utilizado por el duplicador de extracción durante la duplicación para buscar registros nuevos.

Figura 2.



Algo que nos será muy útil son las asignaciones estáticas, que son listas permanentes de nombres y direcciones que no pueden eliminarse ni modificarse, excepto cuando el administrador quite la asignación específica. Éstas asignaciones estáticas son las que nos van a permitir destinar nombres y direcciones destinadas a clientes no WINS.

Debemos evitar depender en exceso de una sola máquina de la red

Para trabajar con las asignaciones estáticas, elegiremos la opción *Asignaciones estáticas,* bajo el menú *Asignaciones.* Para crear una pulsaremos sobre el botón *Agregar asignaciones*, y seguidamente especificaremos un nombre y una dirección IP estática. Luego estableceremos el tipo:

- Exclusivo. Generamos un nombre único en la base de datos con una dirección por nombre. Originará tres entradas en la base de datos con los valores 00, 03 y 20 (en hexadecimal).
- Grupo. Generamos un grupo normal que no tiene una dirección asociada. Originará una entrada en la base de datos con el valor 1E.
- Nombre de dominio. Generamos un grupo NetBIOS que puede almacenar hasta 25 direcciones de miembros. Origina una sola entrada con el valor 1C
- Grupo Internet. Creamos un grupo especial, definido por el usuario, que puede almacenar 25 direcciones junto al nombre del grupo NetBIOS. Genera una entrada con el valor 20.
- Multitarjeta. Especificamos un nombre único que puede tener hasta 25 direcciones. Cada dirección se corresponderá con tantas tarjetas como tenga instalado el equipo. Se crearán las mismas entradas con los mismos valores que con el tipo exclusivo.

Figura 3.

Propiedades de TCP/IP

Enlaces | Avanzado | Configuración DNS
Gateway | Configuración WINS | Dirección IP

Póngase en contacto con el administrador de red para investigar si necesita configurar su PC para WINS.

Desactivar la resolución WINS

Activar la resolución WINS:

Servidor principal WINS: 195.180.123. 1

Servidor secundario WINS: . . .

Identificador de ámbito:

Usar DHCP para resolución WINS

Aceptar | Cancelar

Para terminar con el servidor de WINS, podemos importar las asignaciones desde el archivo LMHOSTS, o de cualquier otro que tenga su mismo formato.

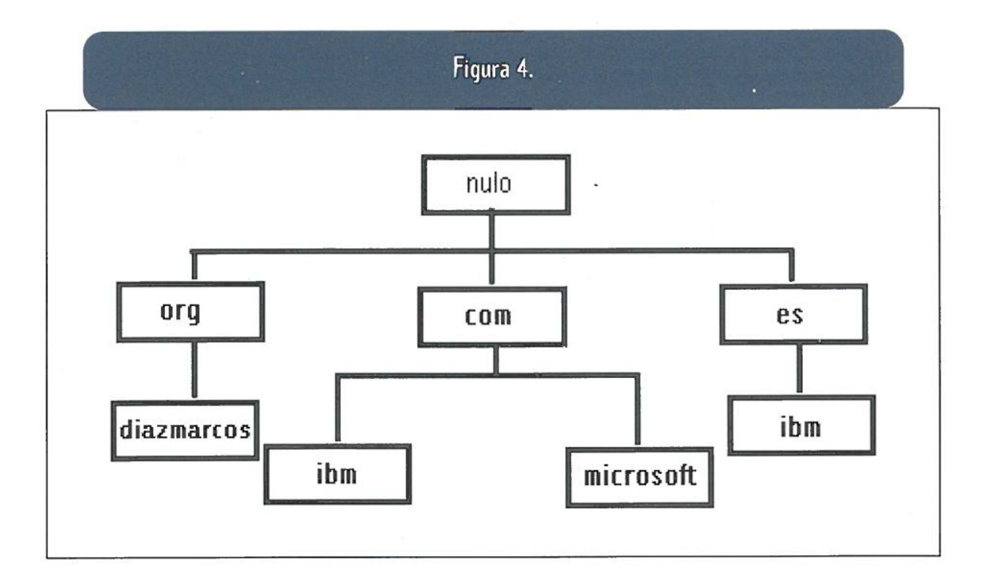
Para que la resolución de nombres NetBIOS funcione perfectamente, sólo nos resta configurar los clientes. Para ello especificaremos en la configuración TCP/IP de los clientes que utilizaremos un servidor WINS, precisando su dirección IP (Figura 3).

Los dominios

Tal como está ahora, nuestra red sólo atenderá nombres NetBIOS, así que implementaremos la de nombres de dominio, el DNS. De este modo nuestros ordenadores serán también accesibles con los nombres de Internet, por ejemplo marivi.diazmarcos.org; y nuestro servidor podrá ser conocido también por estos nombres. En nuestro caso implementaremos al menos dos: dafne.diazmarcos.org y www.diazmarcos.org.

El DNS no es otra cosa que una base de datos distribuida que proporciona un sistema de nomenclatura jerárquico para identificar *hosts* en Internet. Aunque DNS y WINS nos parezcan el mismo perro con distinto collar, existe una diferencia principal: WINS es totalmente dinámico, mientras que DNS necesita una configuración estática para la asignación de nombres de equipos a direcciones IP.

La base de datos DNS es una estructura de árbol (figura 4) denominada espa-



cio de nombres de dominio, en la cual cada dominio (nodo en la estructura de árbol) tiene un nombre y puede contener subdominios. Esta estructura es jerárquica y los nombres de subdominios han de ser únicos dentro de sus dominios. Así, Por ejemplo, el nombre de dominio DNS ibm.es especifica el subdominio ibm cuyo dominio principal es es; ibm.com especifica el subdominio ibm dentro del dominio com. El nombre de dominio identifica la posición del dominio en la base de datos con respecto a su dominio principal y utiliza un punto (.) para separar cada parte de los nombres de los nodos de la red del dominio DNS.

La raíz de la base de datos DNS no tiene nombre, su valor es nulo (*null*). Haremos referencia a él en los nombres DNS con un punto (.) al final. Por ejemplo, en el nombre *www.ibm.es.*, el punto que hay después de com indica el nodo raíz DNS.

En Internet, el organismo InterNIC administra la raíz y los dominios de nivel superior de la base de datos DNS, naturalmente esta administración no es gratuita, y si deseamos salir a Internet deberemos pagar para registrar nuestro nombre en el dominio. Existen tres tipos de nombres superiores de dominio. A saber: los de organizaciones, de tres caracteres (com, edu, etc.); los geográficos, con códigos de país de dos caracteres (es, uk, etc.); y el dominio especial in-addr.arpa que se usa para relacionar nombres y direcciones.

Los nombres de dominio de organizaciones se usaron mucho cuando Internet no era tan grande, e indican el tipo de actividad del propietario del dominio (tabla 3), pero en la década de los 80, con el enorme crecimiento de Internet, se hizo evidente que este tipo de división era inadecuada para una entidad global tan grande. Entonces se introdujeron los nombres de dominio geográfico.

InterNIC administra la raíz y los niveles superiores de la base de datos DNS

InterNIC delega en otras organizaciones la responsabilidad de administrar el espacio de nombres de dominio DNS por debajo del nivel superior. Estas organizaciones posteriormente subdividen el espacio de nombres y delegan responsabilidades. Así, en España RedIRIS se encarga de administrar los dominios del tipo *dominio.es*. Este modelo administrativo descentralizado permite que DNS se administre de manera autónoma en los niveles que sean razonables para cada organización involucrada.

Las zonas constituyen la unidad administrativa en DNS. Una zona es una rama de la base de datos DNS y se administra como una entidad separada. Puede consistir en un único dominio o en un dominio dividido en subdominios. Los subdominios del nivel inferior de una zona también se pueden dividir en una zona o zonas separadas.

Resolución de nombres de dominio

Tener una base de datos con los datos de todos los nodos y dominios de Internet resulta impensable. Por ello, el DNS usa un modelo cliente-servidor donde los servidores DNS (servidores de nombres) contienen información de sólo una parte de la base de datos DNS (normalmente su propia zona) y hacen que esta información esté disponible para los clientes (*resolvers*). Un *resolver* realiza una consulta de un servidor de nombres pidiendo información acerca del espacio de nombres DNS. Este servidor de nombres puede, a su vez, consultar otros servidores del mismo tipo al intentar responder a la consulta realizada por el *resolver*.

La tarea clave de DNS, al igual que WINS, es presentar nombres sencillos a los usuarios y resolver esos nombres en direcciones IP. La resolución de nombres la proporcionan los servidores de nombres a través de DNS, interpretando la información de un nombre completo de dominio (FQDN) para obtener su dirección específica. El proceso empieza cuando un cliente pasa una consulta a su servidor de nombres local. Si el servidor de nombres local no tiene los datos solicitados en la consulta, los solicita a otros servidores de nombres en nombre del cliente.

La resolución de nombres DNS pasa por tres fases: recursividad, iteración y obtención.

- Recursividad. Un cliente pasa una petición de resolución recursiva a su servidor de nombres local. Una petición de resolución recursiva indica al servidor de nombres que el cliente espera una respuesta completa a la consulta, y no un puntero a otro servidor de nombres. La resolución recursiva pasa de manera efectiva la carga de trabajo al servidor de nombres y permite que el cliente no requiera tener grandes recursos.

- Iteración. En caso de que el servidor de nombres local no pueda resolver la consulta por completo, obtendrá la ayuda de otro servidor de nombres DNS. Un servidor de nombres local realiza la tarea de procesado él mismo y pasa sólo la resolución iterativa a otros servidores de nombres. Una petición de resolución iterativa indica el nombre del servidor del que el peticionario espera recibir la mejor respuesta sin la ayuda de otros servidores de nombres. Si el servidor de nombres tiene los datos solicitados, los devuelve; de lo contrario, envía punteros a los servidores de nombres que tienen más probabilidad de tener una respuesta. Un servidor de nombres principal que no es capaz de resolver una petición de datos de su zona, devuelve un error al peticionario.

- Obtención. Al tiempo que los servidores de nombres locales procesan de manera recursiva las peticiones, descubren mucha información sobre el espacio de nombres de dominio DNS. Para aumentar el rendimiento de DNS y facilitar las tareas en el conjunto de redes y en los otros servidores de nombres, estos últimos mantienen temporalmente esta información en la memoria caché local. Cuando llega una petición del cliente, el servidor de nombres local comprueba la información estática y la memoria caché para dar una respuesta. Incluso si la respuesta no está en la memoria caché, la identidad del servidor de nombres de la zona sí que puede estar, lo que reduce el número de peticiones iterativas que el servidor de nombres tiene que procesar.

DNS y WINS: un buen matrimonio

La estructura de una zona DNS cambia si se agrega un nuevo host o cuando uno existente se mueve a una subred diferente. Puesto que DNS no es dinámico, deberíamos cambiar manualmente los archivos de la base de datos DNS de la zona para reflejar la nueva configuración. Esto nos acarrearía una carga administrativa, especialmente en las zonas que cambian frecuentemente. Para evitarlo, podemos utilizar WINS, que está diseñado para llevar a cabo automáticamente estas tareas administrativas.

Al emparejar DNS y WINS aprovecharemos los puntos fuertes de cada uno para conseguir un servidor DNS dinámico. De esta manera haremos que el DNS consulte a WINS la resolución de nombres de los niveles inferiores del árbol DNS en sus zonas. Este enlace se realiza de forma transparente para los clientes DNS, de modo que parece que los servidores de nombres DNS controlan el proceso completo.

Ni el cliente ni ningún servidor tienen por qué saber que es WINS quien ha resuelto la petición

Cuando un cliente DNS pida la resolución de un nombre FQDN a nuestro servidor DNS, éste le pasará al servidor WINS la parte izquierda del nombre, es decir, el *host*. Es así como le pasamos al servidor WINS un nombre NetBIOS para que lo resuelva. WINS devolverá la dirección del IP del host al servidor DNS, que a su vez se la enviará al cliente. Como puede verse ni el cliente ni ningún otro servidor de nombres DNS sabe que WINS a estado involucrado en el proceso de resolver el nombre.

En nuestra red, por ejemplo, tenemos un ordenador cuyo nombre NetBIOS es marivi, y su FQDN es marivi.diazmarcos.org. Si alguien preguntara al servidor DNS por marivi.diazmarcos.org, el servidor DNS pasaría al WINS la parte izquierda del nombre, esto es, marivi. El servidor WINS resolvería el nombre y devolvería al servidor DNS la dirección 195.180.123.12. El servidor DNS se la enviaría a su vez al cliente. Resuelto.

Instalación del servidor DNS

Para instalar el servidor de nombres DNS, ejecutaremos el icono *Red* del *Panel de control,* seleccionaremos la ficha *Servicios* y pinchamos el botón *Agregar.* Seleccionaremos la entrada *Servidor DNS de Microsoft*. Tras la copia de archivos, se nos habrá añadido la entrada *Administrador de DNS* en la lista de *Herramientas administrativas.*

Antes de pasar a configurar el servidor, ejecutaremos de nuevo el icono *Red* del *Panel de control* y, en las propiedades de protocolo TCP/IP, seleccionaremos la ficha *Dirección DNS.* Allí daremos el FQDN de nuestro servidor (Figura 5). En el apartado *Nombre del host* especificaremos el nombre del equipo, en este caso dafne; y en dominio el nombre del dominio al que pertenece nuestro equipo, en este caso diazmarcos.org. Si lo deseamos podemos elegir para *Nombre del host* un nombre distinto al nombre NetBIOS del equipo. De esta forma

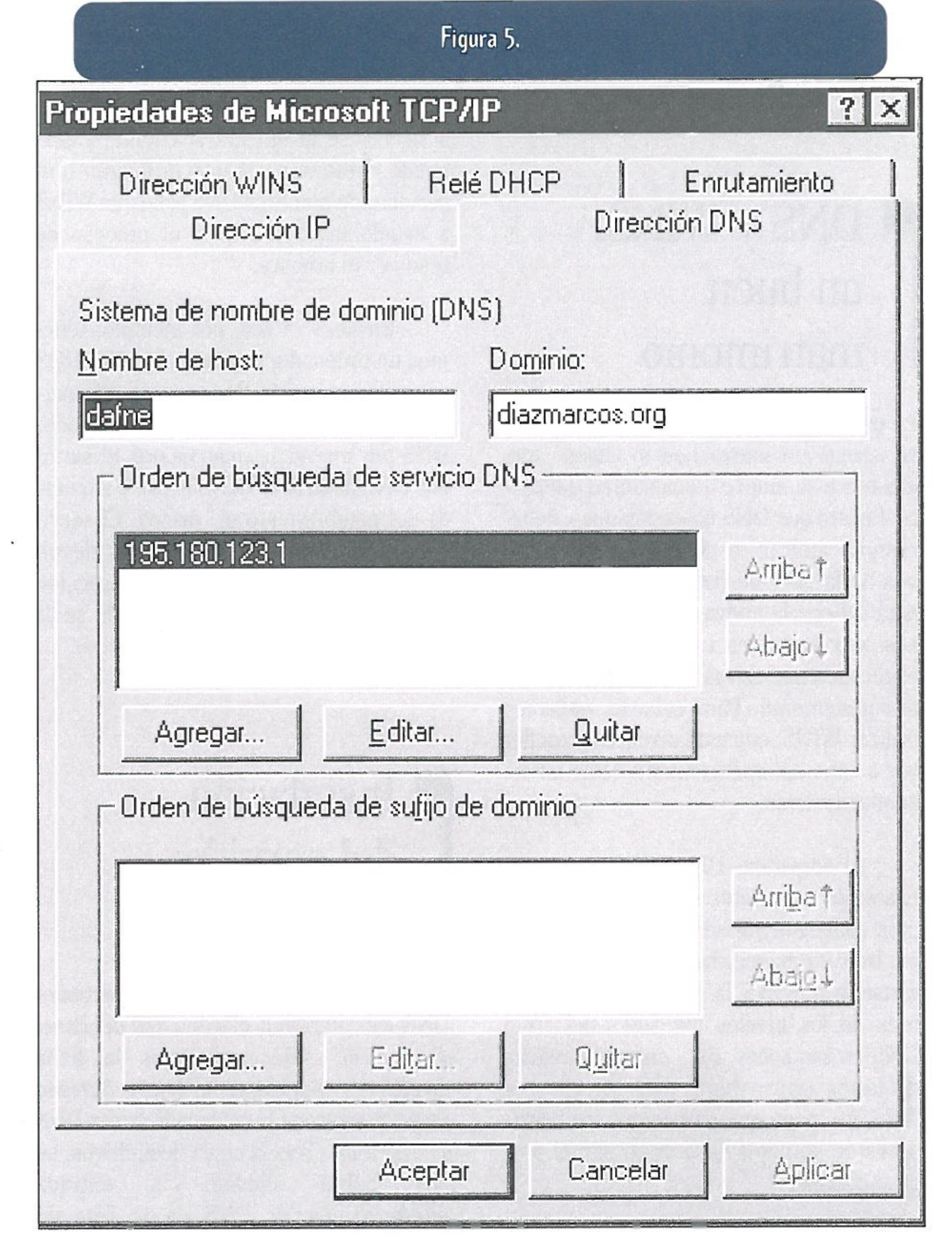


Figura 5.

nuestro ordenador será conocido por nombres distintos en la red local y en Internet. También añadiremos la dirección IP de nuestro equipo al apartado *Orden de búsqueda del servicio DNS*. En este caso 195.180.123.1.

Una vez hecho esto, pasaremos a configurar el servidor DNS. En la lista de Herramientas administrativas ejecutaremos la entrada *Administrador de DNS*. El administrador DNS puede administrar remotamente otros servidores, así que lo primero que haremos es añadir nuestro propio servidor. Para ello seleccionaremos la entrada *Lista de servidores* en el panel izquierdo del *Administrador de servicio de nombres de dominio* y pincharemos la opción *Nuevo servidor* bajo el menú *DNS*. Nos aparecerá un cuadro de diálogo para especificar el nombre o la dirección IP del nuevo servidor. Por ello dimos un nombre FQDN a nuestro servidor en la ficha del protocolo TCP/IP; si no lo hicimos entonces, deberemos hacerlo ahora y reiniciar.

A continuación le daremos el FQDN, nombre completo de dominio, de nuestro servidor, en este caso dafne.diazmarcos.org. Esto es muy importante. Si le damos sólo la parte izquierda del nombre (dafne) le pasaríamos un nombre NetBIOS, y no obtendríamos el mismo resultado. Así pues, le damos el nombre FQDN y pulsamos *Aceptar*. Como vemos, no sólo ha creado el servidor, sino algunas cosas más: los recursos de DNS de la clase *in* (Internet). De momento, no tocaremos nada en los recursos creados automáticamente, pero al que le guste investigar, puede encontrar aquí la razón de la reserve de algunas direcciones IP, como las direcciones de *host* 0 y 255 o la red 127. Eso entre otras muchas cosas.

Crear una zona

En primer lugar, vamos a crear una zona primaria. Para ello seleccionaremos el servidor recién creado en la lista de servidores y pincharemos la opción *Nueva zona* bajo el menú *DNS*, o bien sacando su menú contextual pinchando con el botón derecho del ratón. En tipo de zona especificaremos *Primaria* y pulsaremos el botón *Siguiente*. En la pantalla que nos aparece debemos suministrar un nombre de zona, en este caso diazmarcos.org. Si saltamos al siguiente campo con el tabulador se nos sugerirá un nombre para el archivo de zona, que normalmente será el nombre de la zona más la extensión.dns.

El administrador de DNS puede administrar remotamente otros servidores de nombres

De nuevo comprobamos que se nos ha generado algo más de lo que pedíamos; esto es, los registros NS y SOA para la zona recién creada. El registro NS (servi-

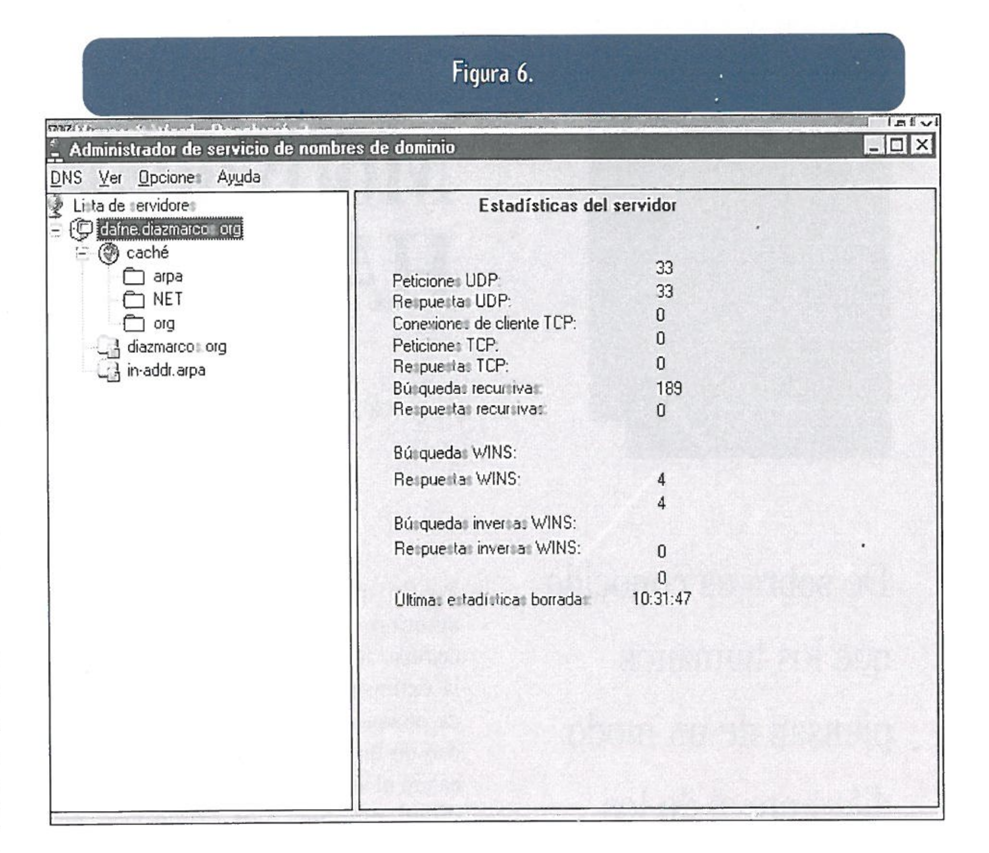
dor de nombres) identifica los servidores de nombres DNS para el dominio DNS. Los registros de recursos NS aparecen en todas las zonas DNS y en las zonas inversas (las del dominio DNS In-addr.arpa). Si hacemos doble clic sobre este recurso, veremos que tiene dos campos:

- Dominio, donde especificamos el dominio al que se aplica este registro.
- Nombre del servidor DNS de nombres, donde especificamos el FQDN del servidor de nombres.

Como es natural, ambos campos se han rellenado adecuadamente y no debemos tocar nada.

El registro SOA (inicio de autoridad) indica que este servidor de nombres DNS es la mejor fuente de información sobre los datos de este dominio DNS. Es el primer registro de cada archivo de base de datos DNS. Aquí se indica de nuevo el FQDN del servidor y otros datos y parámetros configurables, entre ellos el buzón de correo de la persona responsable de la zona y el intervalo de actualización de los datos del servidor.

Figura 6.



El enlace con WINS

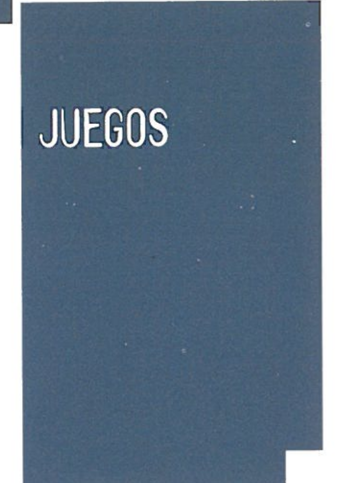
A continuación enlazaremos nuestro flamante servidor DNS al servidor WINS y activaremos la búsqueda en WINS. Para hacerlo seleccionaremos nuestra nuevecita y flamante zona y haremos clic con el botón derecho del ratón para sacar su menú contextual, donde seleccionaremos la opción *Propiedades*. En el formulario de propiedades seleccionaremos la ficha *WINS Lookup*. En ella proporcionaremos la lista de servidores WINS a utilizar. En nuestro caso no tiene pérdida, ya que sólo tenemos uno y está en la misma máquina; así que le damos la dirección 195.180.123.1. También activaremos las casillas *Utilizar la resolución de WINS* y *La configuración sólo afecta al servidor.* Aceptamos y comprobaremos que se nos ha añadido un nuevo registro en nuestra zona: un registro de tipo WINS.

Ya sólo nos queda activar las búsquedas inversas con WINS: Seleccionamos la zona in-addr.arpa, y en sus propiedades seleccionamos la ficha *Búsqueda inversa WINS*. Marcamos la casilla Utilizar búsqueda *inversa WINS* y proporcionamos el nombre del dominio del host DNS. En este caso diazmarcos.org. Este nombre de dominio se combinará con el devuelto por WINS para formar un FQDN. El servidor DNS remitirá la dirección IP de las peticiones DNS a WINS para su resolución final. Si una búsqueda de WINS devuelve dafne, el servidor DNS lo combinará con el nombre de dominio que hemos proporcionado y devolverá al cliente dafne.diazamarcos.org. Si lo hemos hecho todo bien, nos quedará algo parecido a la figura 6. Para terminar, vamos a añadir un detalle: un registro de tipo A (*host*) que se llame www. De este modo nuestro ordenador también responderá por www.diazmarcos.org. Así pues, seleccionamos la zona diazmarcos.org y pinchamos en la opción *Nuevo host* bajo el menú *DNS*. Tecleamos el nombre, www y la dirección IP, 195.180.123.1.

Ya tenemos configurado el servidor DNS. Ahora los ordenadores de nuestra red responderán tanto a nombres NetBIOS como a nombres DNS. Buena muestra de la cooperación entre DNS y WINS es que en la zona diazmarcos.org se van añadiendo *hosts* (registros de tipo A) a medida que WINS va teniendo conocimiento de ellos. En fin, teóricamente ya funciona solo y no debemos ocuparnos mucho de los servidores WINS y DNS.

Próximo número

Una vez implementado el protocolo TCP/IP y la resolución de nombres, el mes que viene instalaremos el Internet Information Server. Nuestra intranet contará con alguno de los servicios básicos de Internet: WWW, FTP y Gopher.



Motivación - tablas HASH

Eugenio Castillo Jiménez

De sobra es conocido que los humanos piensan de un modo diferente al de los programas de ajedrez. Las personas analizan un árbol de jugadas más reducido y cada una de dichas jugadas posee una razón muy determinada para ser considerada. Mediante la motivación, podemos reducir el número de ramas a estudiar.

Por lo que hemos visto en artículos anteriores en un programa de ajedrez el criterio más importante que determina la extensión o crecimiento de un árbol de búsqueda es el del nivel de profundidad de fuerza bruta al que se está realizando el análisis, la denominada **profundidad nominal**. Los extensores, y la búsqueda de capturas o aquiescencia, son nuevas formas capaces de ampliar dicha profundidad dependiendo del tipo de jugadas que se realice, este tipo de influencia de las jugadas sobre el árbol provocaba **incrementos enteros** de la profundidad nominal. Para estos casos podemos afirmar que al igual que los humanos hemos empleado unos conocimientos, aunque primitivos, que modifican el crecimiento del árbol.

Existen un buen número de programas de ajedrez que han utilizado otro tipo de filosofía de extensión a la que denominaremos **motivación**, imaginemos que los incrementos de profundidad pudieran ser fraccionales y que cada jugada/pliegue consumiera 5 puntos de fracción. Si partiésemos de un análisis de 15 puntos podríamos realizar una búsqueda de fuerza bruta equivalente a 3 pliegues. Si le asignáramos un **consumo de motivación variable** a cada jugada según la naturaleza de ésta y continuásemos el árbol hasta agotar la motivación, entonces obtendríamos un árbol con ramas de extensiones diferentes y permaneciendo siempre dentro de la fuerza bruta.

Veamos una posible valoración de las motivaciones de las jugadas con un ejemplo en C (véase listado 1); suponemos que la jugada ya ha sido realizada y que todas las tablas de información accesorias al movimiento se encuentran actualizadas.

Este tipo de valoraciones sólo son un ejemplo, no se han incluido factores de ataque como el de amenazar piezas contrarias o el apoyo del avance de un peón pasado cubriendo las casillas que conducen a su coronación, para poder realizar esto hemos de crear un proceso similar al de cubrepiezas sólo que en este caso las comparaciones han de realizarse al contrario. Hay que señalar que estas comparaciones son muy necesarias puesto que si sólo escrutásemos los objetivos cumplidos pudiera ser que ya hubiera alguno con anterioridad y que el nuevo movimiento generado no produjese ningún tipo de novedad. Para realizar esto correctamente hemos de guardar las referencias de las fuerzas que se ejercen sobre los objetivos antes y después del movimiento.

Bajo las características de ataque sólo hemos incluido las jugadas de jaque por ser las más notorias, aquí no hemos considerado la cantidad de casillas libres que tuviera el rey lo cual en muchas ocasiones constituye un criterio primordial a la hora de seleccionar entre buenos y malos jaques, se hace muy evidente en algunos casos en los que el rey sólo posee

1 ó 2 casillas de escape que el peligro de mate o ganancia de material es muy alto.

Hay motivaciones que pueden variar respecto a la fase del juego en que nos encontremos, bajo este caso tenemos :

- Los peones pasados: su importancia aumenta cuanto más cerca nos encontremos del final de la partida.
- Desarrollo de piezas o enroque: su influencia es mayor en la apertura y medio juego que en el final.
- Ataque al rey: nos referimos al avance de peones y aperturas de columnas para producir ataques a la bayoneta, destacan en el medio juego.

Algunos programadores incrementan de un modo automático el consumo de motivación de las jugadas en caso de que exista una pérdida material considerada como innecesaria en las jugadas precedentes, este incremento es tanto más alto cuanto más se avance en la búsqueda. Un resultado similar se obtenía utilizando el movimiento o turno nulo publicado por Döninger, aunque ambas ideas posean una naturaleza diferente al fin y al cabo ambas decrementan la extensión del árbol.

Existe un efecto negativo que se puede dar con relativa frecuencia y que muestra actuaciones ilógicas (para algunos humanos calificadas como "sentimentales"), estas se vienen a dar cuando existe una larga combinación que agota o deja al límite la motivación asignada para la iteración en curso (algo parecido al **efecto horizonte**). Si el ordenador jugara con el bando perdedor e hiciera jugadas malas adrede (jugadas con un alto consumo de motivación) entonces la línea perdedora no llegaría a identificarse puesto que la podaría antes por falta de motivación, autoocultandose de esta forma la auténtica línea. Esta "estupidez" fue remediada utilizando un **contador de motivación independiente para cada bando**, de este modo si uno de los bandos acababa con su motivación y el otro aún poseía reservas la línea se continuaba y podía dar con la jugada decisiva.

Listado 1

```
/*
short atak [2][64]; //atak [color,sq]
atak posee los valores de fuerza de todas las piezas que llegan a la casilla sq, color (de
valores 0 ó 1) determina el bando, este tipo de tabla es similar a la que emplea el programa GNU.

unsigned char board [64];
board contiene el tipo de pieza que hay sobre cada casilla

short NumAtak [65536];
cada casilla de atak posee 65536 estados posibles y NumAtak dice la cantidad de piezas
de un bando que llegan a la casilla.

short capturas [maxprof];
en cada turno que hubiera capturas se dice la casilla donde se produció, de este modo si
en el turno "ply + 1" se da otra captura y su casilla coincide con la de "ply" entonces se trata de
una recaptura.

short esjaque;
variable que indica si el rey del bando en juego está en jaque.

short valormedio;
valormedio indica un valor considerado como de equilibrio material, muchas veces tiende
a coincidir con el alfa.
*/

short cubrepiezas (color,contrario)

/*cubrepiezas identifica los objetivos del contrario sobre los que ha disminuido la influen-
cia de este o ha aumentado la del propio bando, para ello efectuamos una comprobación del esta-
do actual de la casilla sobre atak con su estado anterior almacenado en la tabla atakprevio. Los
objetivos pueden ser tanto piezas como casillas vacías donde generalmente intentará dirigirse
alguna pieza. La estructura de los objetivos puede ser:

struct tobjetivo { short cant;
         short casilla [35];
      };
*/

{short f;

   for (f = 0;f < objetivo.cant;f++)
   { if (atakprevio [contrario,objetivo.casilla [f]] > atak [contrario,objetivo.casilla [f]] ||
         atakprevio [color,objetivo.casilla [f]] < atak [color,objetivo.casilla [f]])
      return (1);
   }
   return (0);
}

short motiv (origen,destino,color,ply)

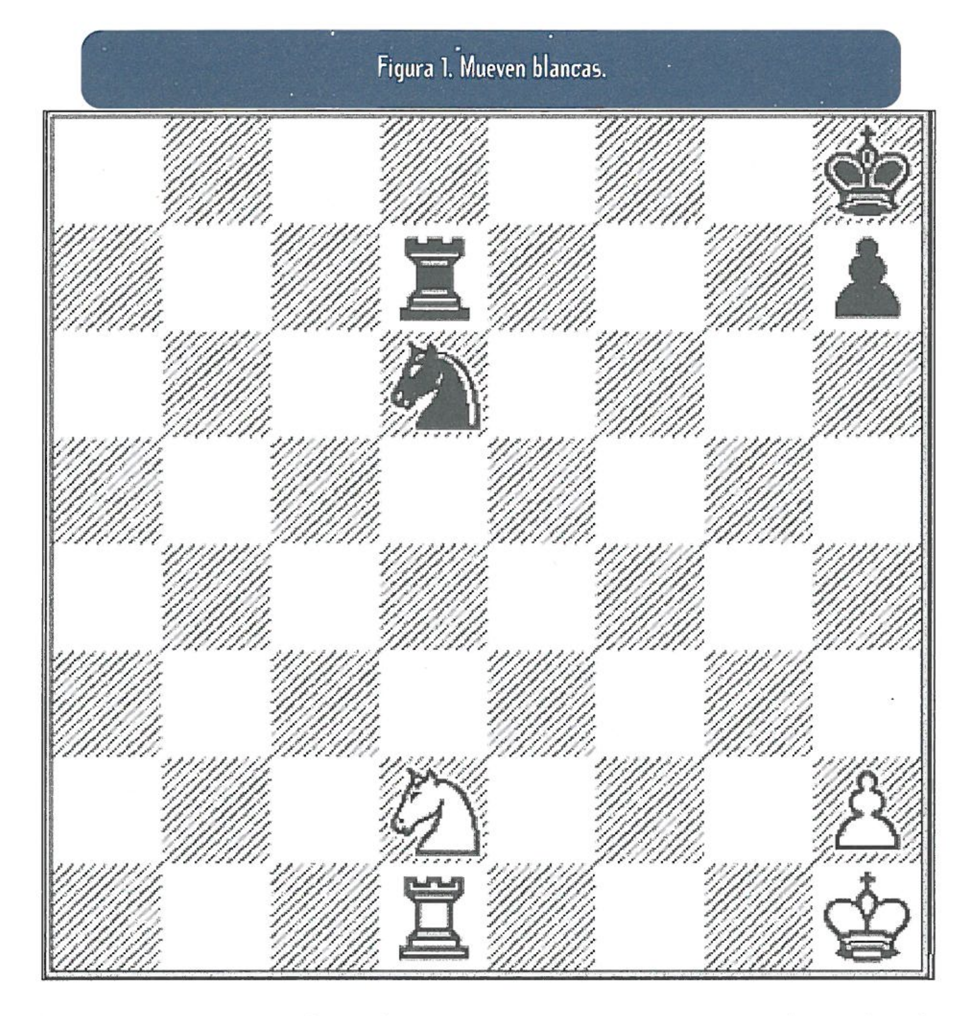
//origen y destino son las casillas de la pieza que mueve, color es el bando al que pertene-
ce y ply es el turno/pliegue de análisis.

{ short s;

   s = 5; //valor inicial del movimiento

  if (esjaque)
     s -= 2;
```

Figura 1. Mueven blancas.



El modo en que continua el juego el bando falto de "motivación" no ha sido publicado pero es de suponer que mediante la utilización de movimientos nulos y movimientos "respuesta" basados en las variantes obtenidas a posteriori. En caso de que el movimiento nulo haya dado una variación negativa mínima respecto al valor inicial de la posición entonces no merece la pena continuar la búsqueda de nuevas jugadas (saltamos directamente fuera del nodo), en caso contrario (el oponente ha encontrado algo significativo) se pueden realizar antijugadas que intenten evitar la línea mostrada como buena para el contrario. El caso de que un turno nulo dé una evaluación altamente positiva es posible, pero carece de sentido puesto que a un turno nulo se le podría contestar con otro igual produciendo una situación estancada, podemos decir que tras cada turno nulo el contrario asume como valor mínimo aceptado el que ya tiene al comenzar su búsqueda, con lo cual evitamos la posibilidad anteriormente mencionada.

Para la generación de antijugadas hemos de seguir unas ciertas normas, las siguientes pueden servir de ejemplo:

1º Amenazar la casilla de destino del siguiente movimiento contrario.
2º Ver si se puede tomar la pieza que va a mover.
3º Intentar clavar o amenazar dicha pieza.
4º Bloquear su ruta de movimiento.

Un caso curioso referido a la generación de movimientos se puede presentar cuando al bando en curso le quede una cantidad de motivación baja y sólo pueda realizar **jugadas objetivos**, es decir aquellas que sean consideradas como buenas puesto que consumen muy poca motivación (de lo contrario son podadas directamente). Podemos ser conscientes de que una jugada va a ser desechada de dos formas: preveyendo su posible consumo o realizando la jugada y calculando entonces su consumo (el ejemplo de código mostrado en "motiv" pertenece al segundo sistema). Los programadores que hayan escogido la primera vía seleccionan a "priori" (mediante algoritmos específicos) reduciendo de este modo el número de jugadas testeadas a posteriori en el árbol, lo cual supone un evidente ahorro. La eficiencia de estos algoritmos siempre tenderá a ser imperfecta puesto que pueden darse un buen número de casos imprevistos que no se corresponden con jugadas patrones, uno de estos casos puede ser el de ña figura 1.

Supongamos que esta posición se presenta tras una combinación dentro de una búsqueda en el árbol, el programa que selecciona las jugadas objetivo a priori ve que el caballo negro de d6 puede ser amenazado por la torre blanca de d1 si el caballo blanco de d2 se retira hacia "cualquier" lugar, también constatará que la amenaza será insuficiente puesto que el caballo de d6 estará cubierto por la torre de d7 negra. Otro tanto ocurrirá cuando vea en una búsqueda paralela que el caballo de por sí también puede amenazar d6 "en solitario", lo cierto es que el movimiento d2e4 del caballo hace que dos piezas ataquen el caballo de d6, siendo esta una jugada ganadora. Si el algoritmo identificador a priori, debido a que no genera las jugadas sino que las supone mediante reglas, no es capaz de relacionar ambos ideas no verá esta posibilidad.

Para la generación de antijugadas es necesario seguir unas normas

Veamos ahora un ejemplo típico en el que la búsqueda por motivación produce efectos devastadores (véase la figura 2).

La línea ganadora del diagrama 2 es 1. Dd3c4+ b5xc4 2. Ac1xc4+ Cb4d5 3.

Axd5+ Te8e6 4. Ad5xe6+ Rg8h8 5. Cf4g6+ Ph7g6 6. Ph5xg6+ Cf5h6 7. Tf2xf8+ Te8xf8 8. Th1h6+ Pg7xh6 9. Ac1b2+ con mate inmediato. Esta combinación puede ser sencilla para un programa basado en la búsqueda por motivación puesto que todas las jugadas del bando blanco son de jaque lo que ya no es tan sencillo es hacer que vea una combinación de tal longitud seleccionando sólo las jugadas más interesantes. Para algunos programas la jugada de jaque 6. Ph5xg6+ puede resultar imposible de encontrar (siempre hablando fuera del campo de la fuerza bruta) debido a que el jaque no lo realiza la pieza que mueve sino la torre que hay detrás, a este tipo de movimiento se le conoce con el nombre de descubierta.

Como podemos ver esta teoría de juego está íntimamente ligada a la de los sistemas basados en el movimiento selectivo su principal diferencia radica en que en estos últimos el incremento fraccional de depth no es una característica obligatoria. Personalmente opino que superan a los programas basados sólo en sistemas de extensores como los mostrados en otros artículos anteriores, también es cierto que requieren un tiempo de desarrollo mucho mayor.

Entre los primeros programas que comenzaron a utilizar esta teoría podemos encontrar al CYRUS 68K y el CHESS MK V de principios de los 80, Mark Taylor es uno de sus programadores, David Levy es también reconocido como uno de los pioneros teóricos. Un exponente ejemplar actual es el programa israelí JUNIOR vencedor del campeonato del mundo de 1997 celebrado en París. Según sus autores, Amir Ban y David Bushinski, su programa da un valor de motivación entre 0 y 3 a cada jugada y el valor de cada turno es de 2 (en vez de cinco como hemos hecho en el ejemplo), también me aseguraron no utilizar el turno nulo. Pude constatar personalmente cómo su creación mostraba líneas de análisis coherentes de más de 6 jugadas (12 plys) antes de 8 segundos, los resultados de este programa hablan por sí solos, 9 victorias y un

Listado 1 (continuación)

```
if (atak [color,reycontrario]) // se está dando jaque al rey contrario
{ s—;
  if (board [destino] == dama)
    s -= 1;
  if (atak [color,destino] >= atak [contrario,destino])
    s -= 1; //la casilla está dominada por el bando que mueve
  if (NumAtak [atak [color,reycontrario]] > 1)
    s -= 1; //premio por jaque doble
}

if (board [destino]) // se trata de una captura
{ if (capturas [ply - 1] != destino) //con esta pregunta podemos saber si se trata de una recaptura
  { switch capturafavorable (origen,destino) {
      case favorable      : s-= 3; break;
      case neutra         : s—; break;
      case desfavorable   : s += 2;
  }
  else //se trata de una recaptura
  { if (valor [board [destino]] + score <= valormedio)
        s += 2; // la jugada no equilibra el material perdido
    else
        s -= 3;
    if (distancia (destino,reycontrario) < 3)
        s -= 2;
    /*en muchos casos las piezas cercanas al rey actuan de escudo frente a los ataques enemigos, por lo que las capturas de dichas piezas son importantes.*/
  }
}
else //no es una jugada de captura
{ if (esenroque (origen,destino))
    return (3); //llevamos el valor directamente fuera

  if (espeonpasado (destino))
  { if (atak [color,destino] >= atak [contrario,destino])
      s -= 2; //el peon pasado no está amenazado
    else
      s ++;
  }

  if (atak [color,destino] >= atak [contrario,destino])
    s += 2;

  if (PiezaHuida [destino])
    s -= 2;

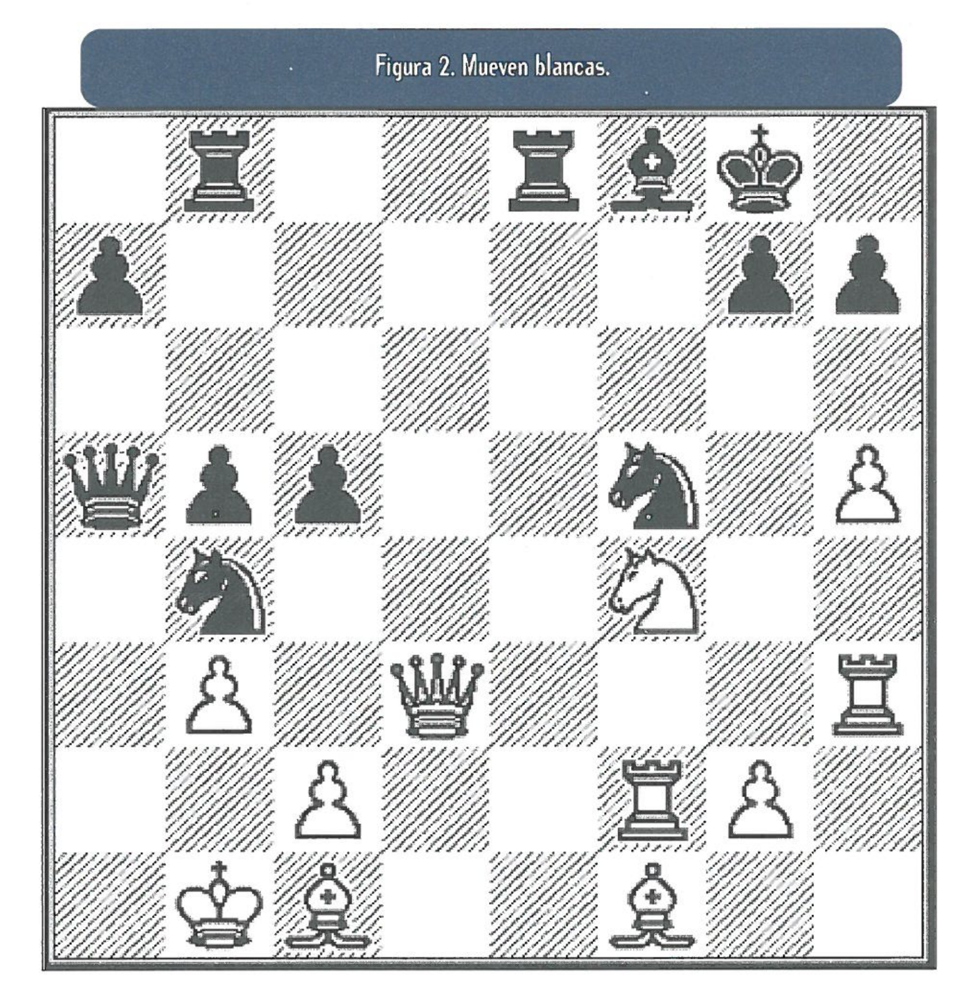
  if (cubrepiezas (color,contrario))
    s -=2;
}

if (s < 0)
  s = 0;

/*el poner s a 0 si cayera por de 0 es algo completamente opcional inherente al estilo de juego que desee alcanzar el programador. Hacerlo de este modo es lo más común.*/

return (s);
}
```

Figura 2. Mueven blancas.



empate de 11 enfrentamientos. También el CRAFTY según su autor Robert Hyatt, utiliza en sus últimas versiones un sistema parecido al menos para el caso de los jaques.

Es evidente que el mantenimiento de todas las tablas de información necesarias y las llamadas a determinadas rutinas enlentecen considerablemente el proceso de generación y evaluación de nodos. Esto dependerá por completo del volumen de datos que deseemos manejar y del modo en que los modifiquemos. Por las opiniones contrastadas con otros programadores parece obligatorio el uso de tablas atak similares a las usadas por el GNU, pero algunos de ellos no rellenan todo el tablero con los campos de fuerza de las piezas sino sólo los lugares ocupados por otras piezas y algunos objetivos, con lo cual reducen el tiempo de escritura para cada nueva posición.

Como ejemplo de casos más extremos de polos opuestos a nivel de velocidad podemos encontrar al CHESS SYSTEM TAL (de Chris Withington) a 6.500 nodos por segundo de media y al JUNIOR con 150.000, ambos programas muestran señales evidentes de usar sistemas de juego selectivo.

Tablas Hash

Las Tablas Hash se han convertido en un componente fundamental de los programas de ajedrez actuales, su efecto más inmediato es el de la simplificación del árbol de análisis gracias a que recuerdan posiciones anteriores.

Utilicemos como ejemplo una estructura de tabla ligeramente diferente a la del GNU

```
struct hashentry
{ unsigned long hashbd;
      //clave encriptada de la posición del tablero.
  unsigned short mv;
      //contiene el movimiento considerado como
      mejor en la posición.
  unsigned char flags, depth,color;
      //depth es la profundidad de análisis restante
      y color el bando que toca mover.
  short score,alfa;
      //score y alfa se corresponden con sus res-
      pectivos del nodo en curso.
};
```

Cuando una jugada "MovX" ha sido elegida como la mejor en un nodo concreto del árbol, nos sería de gran ayuda almacenar esta información por si en la siguiente iteración nos volviéramos a topar con la misma posición. En este caso comenzaríamos el análisis por la jugada "MovX" con lo que conseguiríamos un mínimo de ahorro en nodos garantizado, es más si la profundidad de análisis restante (depth) fuera igual o inferior a la de MovX entonces ni tan siquiera necesitaríamos reanalisis puesto que se trataría de una transposición (llegamos desde distintos lugares al mismo sitio), en este caso devolvemos como jugada resultado a MovX y como valor a score.

Existen trasposiciones que no pueden ser tomadas como válidas y son aquellas que parten de hipótesis diferentes, en este caso con hipótesis me refiero al valor del alpha de la jugada MovX, si éste fuera inferior al alfa en curso o superior al beta no podemos tomar el score obtenido como un valor de confianza. Estos casos se nos pueden presentar ante todo cuando se efectúan reevaluaciones a una misma profundidad pero con diferentes alfa y beta.

Veamos, mediante un ejemplo, cómo se introduciría la información de una jugada en las Tablas Hash (véase listado 2).

La comprobación *1* es muy importante, la variable flag.timeout indica si ya hemos pasado la cantidad de tiempo asignada para pensar si este fuera el caso hemos de tener cuidado con los valores que devolvamos para que no se introduz-

can en las tablas hash puesto que son valores que o no se han terminado de comprobar o se han tomado tal cual sin más.

El hecho de haber incluido el color del bando que mueve es muy importante, en versiones antiguas del GNU no se consideraba esto y se podían producir situaciones embarazosas, pues atribuía valores blancos a posiciones negras y viceversa.

Las variables hashkey y hashbd representan una posición encriptada, en *2* podemos ver cómo se busca una localización para la jugada. En caso de no existir una coincidencia (posición anterior análoga) se busca un lugar que se pueda sobreescribir intentando no escogerlo con un ptbl->depth > depth. rehash es la constante que indica el número de reintentos que se pueden hacer antes de dar por terminada la búsqueda.

Es muy importante incluir el color del bando que se mueve

Existen algoritmos más sofisticados que incluyen la **antiguedad** de la referencia hash introducida, o la cantidad de veces que se ha hecho uso de ella, incluso algunos que guardan colas de elementos borrables. El hecho es que dado que el número de nodos analizados en pocos segundos podría llenar una tabla con miles de elementos, es necesario disponer de un mecanismo que simule la capacidad de "olvidar" variantes anteriores que tengan una menor probabilidad de ocurrir quedándose de este modo con aquellas que tiene mayor valor.

En *3* podemos ver el uso de flag para clasificar las jugadas introducidas comparándolas con su alfa-beta, la que mayor consideración merece por su fiabilidad es la calificada como truescore. El caso upperbound lo he incluido como ejemplo pero no conozco a nadie que lo use puesto que cada vez que se llama a PutInTTable se comprueba que "best > alfa" (best equivale a score), con lo que este caso jamás se daría.

Listado 2

```
#define ttblsz = 1 << 12; //el tamaño de la tabla dependerá de nuestra memoria

struct hashentry ttable[2][ttblsz];

void PutInTTable (long color,
            long score,
            long depth,
            long alfa,
            long beta,
            long mv)

{ struct hashentry *ptbl;
  long i;

 if (flag.timeout) //**1**
   return;

 ptbl = &ttable[color][hashkey & (ttblsz - 1)];

for (i = 1; depth < ptbl->depth && ptbl->hashbd != hashbd && i <= rehash; y++) //**2**
  ptbl = &ttable[color][(hashkey + i) & (ttblsz - 1)];

 if (depth >= ptbl->depth || ptbl->hashbd != hashbd)
 { ptbl->alfa = alfa;
   ptbl->hashbd = hashbd;
   ptbl->depth = (unsigned char) depth;
   ptbl->score = score;
   ptbl->mv = mv;
   ptbl->flags = 0;
   ptbl->side = side;
   if (score < alfa) //**3**
       ptbl->flags |= upperbound;
   else if (score > beta)
       ptbl->flags |= lowerbound;
   else
      ptbl->flags |= truescore;
#ifdef HASHTEST //**4**
     for (i = 0; i < 32; i++) {
        ptbl->bd[i] = CB(i);
     }
#endif /* HASHTEST */
  }
}
```

La medida de seguridad de *4* es costosa en tiempo de proceso y sólo le es útil a los "hipocondríacos" de las colisiones, si lo comprobamos podemos observar que hashkey y hashbd no pueden representar todas las posiciones posibles de un tablero, por lo que la pro-

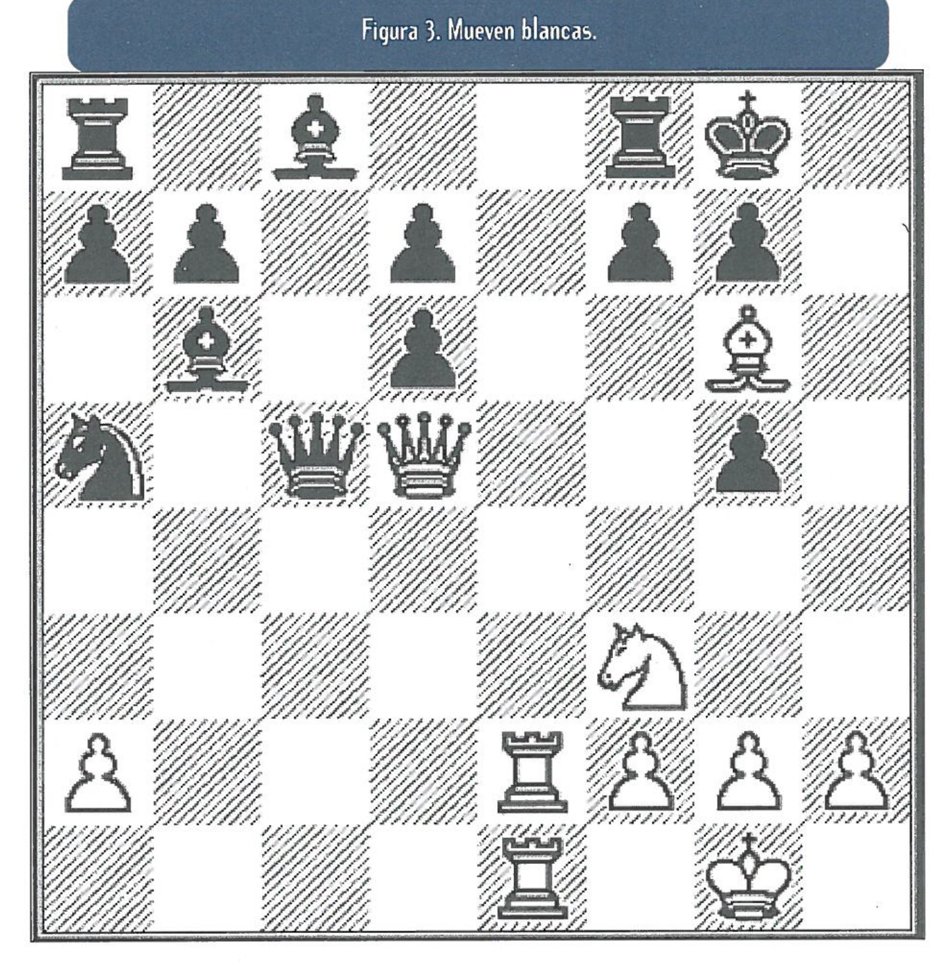


Figura 3. Mueven blancas.

babilidad de que exista una confusión (que dos posiciones tengan la misma clave) existe. Este es un riesgo mínimo que corren todos los programas de ajedrez que conozco.

Cabría suponer que cuanto mayor fuera la tabla hash mejores resultados se obtendrían, pero ello no es cierto el número de entradas que yo suelo usar es de 1 << 16, una cantidad superior no me ha producido ningún efecto notable, por lo que pudiéramos señalar a este número como un tope útil.

Tabla 1. Direcciones de internet donde obtener información:

Si desea obtener más información, puede acceder, a través de Internet, a las siguientes direcciones:

www.icdchess.com
www.chessformat.com
(web en español)

El uso de estas tablas no se restringe tan sólo al árbol de análisis, también podemos sacarles provecho intentando acelerar las evaluaciones que en muchas ocasiones no realizan más que miles de veces procesamientos altamente similares que podrían ser obviados acumulando algunos de los patrones genéricos que menos varían. Unos de los casos más obvios es el de la estructura de peones si ésta no cambia no es necesaria su reevaluación (esto es muy discutible puesto que en un final la simple posición del rey cambia drásticamente una situación) basándose en este criterio muchos programas almacenan las estructuras y no las reevaluan (o no lo hacen al menos en su totalidad previniendo lo anteriormente comentado acerca de la posición del rey). Se han intentado otros casos similares de aprovechamientos pero el de los peones parece ser el más rentable, los demás son casi despreciables por su beneficio.

Más de una persona al ver los efectos inmediatos de una tabla hash la ha asociado con un sistema de aprendizaje básico. Esto pudiera ser considerado como cierto si se resolvieran las siguientes cuestiones:

1. Almacenar las tablas sin perder los datos cada vez que apagásemos el ordenador, manteniendo un criterio de elementos borrables cuando hubiera problemas de memoria.

2. Crear un sistema de selección de jugadas "buenas" mejores que los basados en el criterio de "best > alfa" que provocaría una saturación inmediata, a no ser que empleásemos masas ingentes de memoria (del orden de varios gigabytes) que pudieran además ser manejadas en RAM.

3. Si introdujésemos tipos de datos diferentes a las puras posiciones exactas a reconocer, entre estos nuevos tipos de datos nos encontraríamos con zonas acotadas del tablero, tipos de posiciones por piezas en juego (especialmente para finales), etc. Y que además de darnos una jugada clave o valoración directa de una posición nos pudiera dar una evaluación parcial a añadir a la ya efectuada.

Muchos programas como el ZARKOV, COMET y GENIUS 5 poseen sistemas de aprendizaje basados en las tablas hash, por las últimas estadísticas realizadas en miles de partidas (en la lista PLY sueca) el Genius5 parece tener el mejor algoritmo de aprendizaje, aunque este factor dista mucho de ser decisivo en la actualidad. Otra cuestión a tener en cuenta es que el aprendizaje empleado no distinga el tipo de rival con el que se juega, pues existen programas rivales para los que una jugada mala es más perjudicial que una buena, esto es cierto incluso en el juego entre humanos, de hecho el

"humano" Lasker (uno de los mejores jugadores de la historia) realizaba a propósito jugadas consideradas como erróneas, sin ser perdedoras, para "sacar de quicio" a sus contrarios.

La acumulación de experiencia

En las partidas de ordenadores la acumulación de experiencia hash de una jugada a otra puede provocar un incremento en potencia de hasta un 30% en un programa. Curiosamente si retrocedemos una jugada y la volvemos a pensar el incremento puede llegar a ser superior, y si volvemos a pensarla, de nuevo incrementaríamos el efecto de los hash. Mediante sucesivas pruebas llegaríamos finalmente a un tope de saturación en rendimiento que variaría con respecto a cada posición.

En la posición del diagrama 3 las blancas dan mate después de jugar 1. Dd5xf7+, mi programa en un nivel normal resuelve la posición tras analizar 10.697 nodos. Si vuelvo atrás la posición lo vuelve a resolver generando 8.297 nodos, si repito el proceso por tercera vez lo hace en 4.145 nodos y a partir de aquí ya no se da ninguna mejora. Este efecto se produce gracias a las tablas hash. Parece evidente pues que mediante sucesivas mejoras enfocadas ante todo al tipo de información que se introduce, en vez de tratar con posiciones exactas si se dan sólo los detalles más importantes, significativos/descriptivos de una posición como el tipo de estructura de peones, y posicionamientos genéricos de piezas mayores, entonces la efectividad de las tablas sería mayor puesto que se reduce el volumen de variedad (resumen por similitud) y las coincidencias serían mayores.

Es evidente que una persona mecaniza procesos motores o intelectuales de una forma parecida sólo que a un nivel de expresión superior, el conseguir igualar o mejorar esto es un reto que aun nos queda por superar.

Bibliografía:

- Goetsch, G. and Campbell, M. (1988). Experiments with the Null Move Heuristic in Chess. Poceedings AAAI Spring Symposium on Computer Game Playing.
- Berliner, H.J. (1979). The B* Tree Search: A Best-first Proof Procedure. Artificial Intelligence, 12, pp. 23-40.
- Christian Döninger (1993). Null Move and Deep Search, Selective-search Heuristic for Obtuse Chess Programs. ICCA Journal September 1993.

Los documentos ActiveX en Visual Basic

Jordi Agost Moré
agost@eup.udl.es

Dentro de todas las novedades del VB5.0, quizás una de las más atrayentes sea la posibilidad de crear documentos ActiveX, y poderlos leer mediante un navegador de Internet, dotando así a nuestro programa de unas posibilidades ilimitadas en cuanto a comunicación se refiere.

Una de las últimas novedades, quizás la más impresionante que nos incorpora el Visual Basic 5.0, es la posibilidad de crear documentos ActiveX. Los documentos active son aquellos formularios que pueden aparecer en ventanas en los exploradores de Internet. Los documentos ActiveX de Visual Basic incorporan la presentación, los hipervínculos y la negociación de los menús.

Para diseñar un documento ActiveX, lo haremos de la misma forma con la que diseñamos un formulario en Visual Basic, por lo tanto del mismo modo que éstos últimos, los documentos ActiveX, podrán contener objetos insertables, y mostrar cuadros de diálogo y formularios secundarios. Los documentos ActiveX también podrán aparecer en el cuaderno de Microsoft Office y podremos escribir código para guardar los datos del documento en los archivos de datos del cuaderno.

Si empezamos a crear documentos ActiveX vamos a tener una serie de ventajas tales como poder aprovechar el entorno de VB para su desarrollo, además suponiendo que deseemos publicar en Internet y que sepamos que el futuro usuario puede leerlo mediante Internet Explorer, lo podemos hacer con documentos ActiveX, con lo cual nos ahorraríamos el trabajo que supone aprender otro lenguaje como puede ser HTML para la publicación en páginas web. Además al contrario que una codificación hecha con HTML, los documentos ActiveX nos dan una información visual inmediata sobre la disposición de los elementos en pantalla. Otra ventaja sería que dichos controles admiten el método AsyncRead, con el que podemos empezar una transferencia asíncrona de datos, lo que nos permitirá ejecutar otro tipo de código mientras obtenemos la información que deseamos.

Una de las últimas novedades de VB 5.0 son los documentos ActiveX

Cuando en VB creemos un documento ActiveX, éste siempre estará compuesto de un objeto UserDocument y de los controles que coloquemos dentro de este objeto. Los objetos UserDocument se almacenan, al igual que los formularios, en archivos de texto, los cuales contendrán el código de origen, las propiedades del objeto mismo y de los controles que hayamos colocado en él. Dicho archivo tendrá la extensión .dob. Aunque si vamos a utilizar elementos gráficos entonces VB los almacenará en un archivo .dox (sería similar a los frx de un formulario).

Cuando se compila un documento ActiveX, VB creará el correspondiente archivo .exe o .dll, pero además observaremos como también se crea un archivo

.vbd, que se guardará en el mismo directorio que el componente compilado, archivo, que es el que interpretará Internet Explorer en nuestro ejemplo posterior.

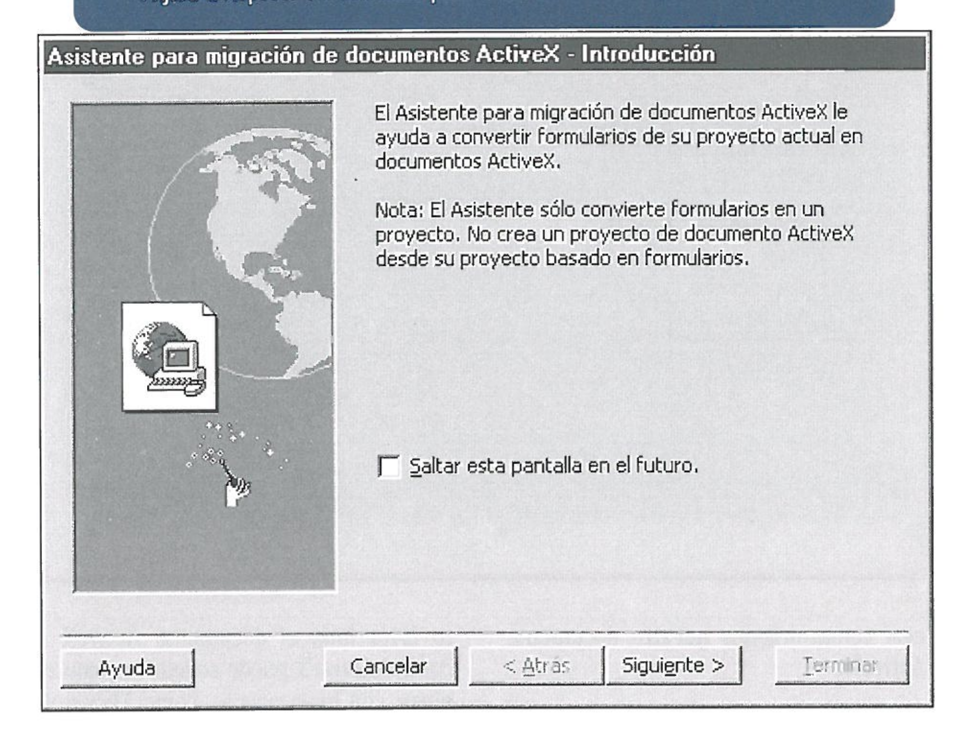
Posteriormente veremos un ejemplo de creación de un control ActiveX, pero debemos saber que si ya tenemos una aplicación hecha y deseamos convertirla en un documento ActiveX, lo podemos hacer mediante el Asistente de migración a documentos ActiveX, cuyo aspecto es el que vemos en la figura1. Para utilizar dicho asistente en primer lugar abriremos el proyecto que deseamos convertir, posteriormente en el menú complementos hacemos un clic en administrador de documentos y seguidamente marcamos la casilla de verificación del "Asistente de migración a documento VB ActiveX", inmediatamente en el menú complementos aparecerá el "Asistente de migración a documentos ActiveX" y lo único que tenemos que hacer es seguir sus instrucciones. Internamente, dicho asistente lo que hará es lo siguiente: en primer lugar copiará las propiedades del formulario a un objeto UserDocument. En segundo lugar copiará todos los objetos del formulario al objeto UserDocument, y mantendrá sus nombres. En tercer lugar, el asistente, comentará todo el código que no es válido (como por ejemplo la instrucción end). En cuarto lugar cambiará el tipo de proyecto a exe o dll ActiveX. Por último, copiará las partes de código que se puedan adaptar, por ejemplo si en nuestro formulario tenemos el evento click, copiará el código al evento click del objeto UserDocument.

El asistente de migración ActiveX nos permite convertir una aplicación a controles ActiveX

Podemos llegar a decir que los documentos ActiveX son semejantes a los controles ActiveX (los controles que normalmente añadimos a un formulario). Quizás la principal semejanza es que un documento ActiveX, al igual que un control ActiveX, necesita un contenedor. En el caso de un control ActiveX dicho contenedor sería el formulario mismo, y en el caso de un documento ActiveX, el contenedor sería Internet Explorer o el Cuaderno de Microsoft.

No obstante, hemos de pensar siempre que los documentos ActiveX, no son documentos aislados, sino que dependen de otras aplicaciones, llamadas contenedores, y que estas pueden ser diferentes, por lo que no siempre podremos tener en cuenta todas las capacidades del contenedor. Otra cosa que hemos de tener en cuenta también es que la forma de mostrar un documento ActiveX, puede variar de un contenedor a otro, por ejemplo, en el entorno de desarrollo integrado (IDE de VB) deberemos utilizar la función CreateToolWindow, pero para mostrar el mismo documento ActiveX en Internet Explorer tenemos que utilizar el método NavigateTo del objeto Hyperlink, junto con el archivo vbd (no requerido por el IDE del VB).

Figura 1. Aspecto del asistente para la traducción de documentos ActiveX.



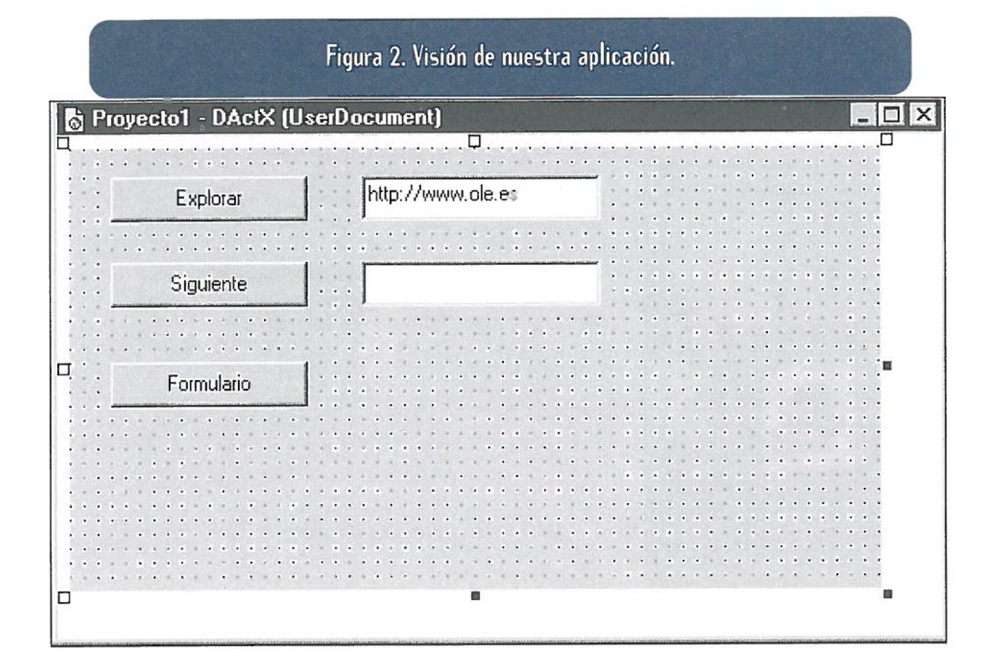
Además otra de las limitaciones con que nos encontraremos a la hora de diseñar documentos ActiveX es que no va a ser posible colocar contenedores OLE (como por ejemplo un documento de Microsoft Word o Microsoft Excel) dentro de un contenedor.

Creación de un documento ActiveX

Vamos a empezar creando un sencillo documento para ver los principios básicos de un documento ActiveX.

Para crear un documento ActiveX, lo primero que haremos es crear un nuevo proyecto de tipo "EXE de documento ActiveX" (Menú Archivo + NuevoProyecto) VB agregará automáticamente una pantalla inicial de documento ActiveX. A esta pantalla inicial, se le denomina diseñador ActiveX, y sería como la base o el formulario a partir del

Figura 2. Visión de nuestra aplicación.



cual construiríamos nuestro documento ActiveX.

En este momento debemos cambiar el nombre del diseñador a DactX, modificando la ventana de las propiedades (propiedad Name). Observemos también que al mismo tiempo, la barra de título o caption del diseñador ha cambiado su contenido, pasando ahora a tener el nombre del diseñador.

A continuación, vamos a colocar en el diseñador un botón (CommandButton) y una caja de texto (TextBox). Modificaremos las propiedades del botón, y así la propiedad Name pasará a ser cmdNavegar y la propiedad Caption será "Explorar". Y de forma análoga, haremos lo mismo con las propiedades de la TextBox, poniendo en su propiedad Text "http://www.ole.es" y en su propiedad Name "txtURL".

Al evento click del CommandButton cmdNavegar le añadiremos el siguiente código:

```
Private Sub cmdNavegar_Click()
  Hyperlink.NavigateTo txtURL.Text
      End Sub
```

En el documento ActiveX y mediante las propiedades y los métodos del objeto Hyperlink, el documento ActiveX o control ActiveX puede solicitar un contenedor con hipervínculo, como Microsoft Internet Explorer, para saltar a una dirección URL determinada. Utilizamos el método NavigateTo para saltar a una dirección de este tipo.

Como nota importante debemos destacar que en la TextBox debe aparecer todo el texto de la dirección que deseemos explorar, incluyendo el protocolo HTTP.

Guardaremos ahora el proyecto, VB guardará el documento ActiveX con extensión .dob y el proyecto con la extensión .vbp

Como ver el documento ActiveX

Para ver el documento que hemos creado en primer lugar deberemos tenemos que ejecutar nuestro proyecto y posteriormente ejecutar una aplicación contenedora con la cual podremos ver dicho documento.

Ejecutaremos Internet Explorer (incluido con VB5), aunque recordemos que también pueden existir otros contenedores como el cuaderno de Microsoft Office, y vamos a explorar un archivo vbd, el cual se ha creado automáticamente al ejecutar nuestro proyecto. Normalmente dicho archivo estará en el directorio predeterminado donde se instalo VB (c:\Archivos de Programa\DevStudio\VB\DactX.vbd"). Si creáramos un archivo exe, entonces el archivo .vbd se localizaría en el directorio de nuestra aplicación. En la figura 3 observaremos el aspecto que tiene el Internet Explorer con nuestro documento ActiveX. Si lo que deseamos es detener la ejecución, añadiremos al código:

```
Private Sub cmdNavegar_Click()
  Stop
  Hyperlink.NavigateTo txtURL.Text
      End Sub
```

Si queremos ejecutar el programa, deberemos ir obligatoriamente a Internet Explorer para ver el documento que hemos creado, y haciendo clic sobre el botón, podemos hacer una exploración del web, cuya dirección hayamos puesto en la TextBox.

Añadiendo un segundo documento

De momento ya hemos logrado poner un documento ActiveX en el explorador de Internet, lo que sucede es que dicha opción es muy limitada, ya que nos ciñe a una sola pantalla, para solucionar este problema vamos a añadir un segundo formulario.

Añadimos en el diseñador DactX un nuevo botón con la propiedad Name igual a "cmdSiguiente" y la propiedad Caption igual a "Siguiente". Y luego en su evento click añadimos el siguiente código:

```
Private Sub Command1_Click()
  Hyperlink.NavigateTo "c:\Archivos de
        Programa\DevStudio\VB" &
        "\dosdActX.vbd"
End Sub
```

Con lo que el aspecto de nuestro documento quedará tal y como vemos en la figura2. Recordemos que si la aplicación estuviese compilada en vez de poner este directorio, deberíamos poner el directorio "app.path & "\dosdActX.vbd". Ahora vamos al menú proyecto y insertamos un segundo documento ActiveX, escogemos "Agregar menú de usuario" y en él elegimos documento de usuario para agregar un segundo documento ActiveX a nuestro proyecto. A continuación vamos a la página de propiedades del documento ActiveX y ponemos la propiedad Name a dosDActX, con lo que también cambiará la barra de título del diseñador, tal y como hemos dicho anteriormente. Una vez hecho esto, vamos al diseñador del segundo documento ActiveX y le añadimos un control de tipo label y un control de tipo CommandButton, a éste último le ponemos la propiedad Caption igual a "Anterior" y la propiedad Name igual a "cmdAnterior" y a la TextBox le ponemos la propiedad name igual a "lblInfo" y la propiedad Caption a "dosDActX"

El menú proyecto permite agregar formularios a nuestra aplicación con un simple click

Añadimos el siguiente código al evento click del botón cmdAnterior:

```
Private Sub cmdAnterior_Click()
  UserDocument.Hyperlink.GoBack
End Sub
```

Donde GoBack es un método del objeto Hyperlink, que nos permitirá ir hacia atrás para explorar el documento anterior del explorador. Seguidamente guardamos el proyecto y a continuación lo ejecutamos y observamos como podemos desplazarnos por los dos documentos.

Agregar un formulario al proyecto

Para agregar un formulario a nuestro proyecto, únicamente debemos ir al menú proyecto, y hacer agregar formulario y establecer las propiedades Name a "frmTemp" y la propiedad Caption a "Formulario de prueba", en el mismo formulario ponemos un Text Box y establecemos sus propiedades Multiline a true, Text igual a "" y la propiedad Name igual a txtInfo. Ahora vamos a DactX , que es el primer documento que hemos creado y le añadimos un botón que será el que nos mostrará el formulario, estableciendo su propiedad Name a cmdForm, y al botón le añadiremos el siguiente código:

```
Private Sub cmdForm_Click()
  frmTemp.txtAux.Text = txtURL.Text
  frmTemp.Show vbModal
End Sub
```

Con lo que cada vez que ejecutemos veremos el formulario.

Una nota a destacar, si nos fijamos en la segunda línea ("frmTemp.Show vbModal") es que mostramos el formulario de forma modal, con lo que se mostrará por encima de cualquier otra ventana de la aplicación. Dicho funcionamiento, que es el mismo que en cualquier formulario de VB, no podemos aplicarlo a un documento ActiveX ya que es el contenedor (en nuestro caso Internet Explorer) el que decide si mostrar el documento o no, y cuando debe hacerlo.

Agregar una propiedad al objeto UserDocument

Dentro del objeto UserDocument podemos, crear propiedades públicas para que otras aplicaciones puedan leerlas o en su defecto establecer valores para dichas propiedades. Es decir podemos escribir o leer propiedades.

Figura 3. Aspecto de Internet Explorer explorando nuestro documento.

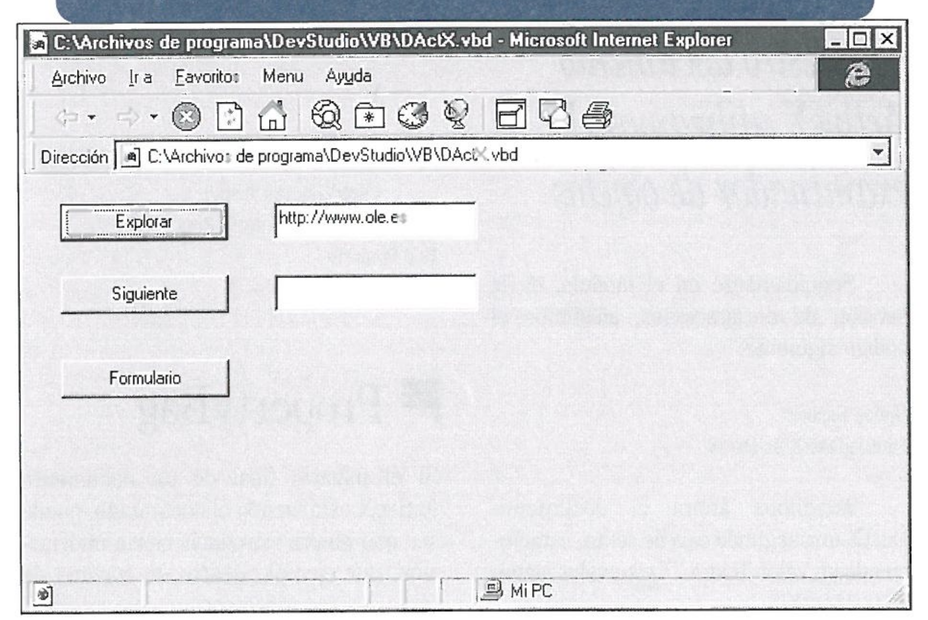
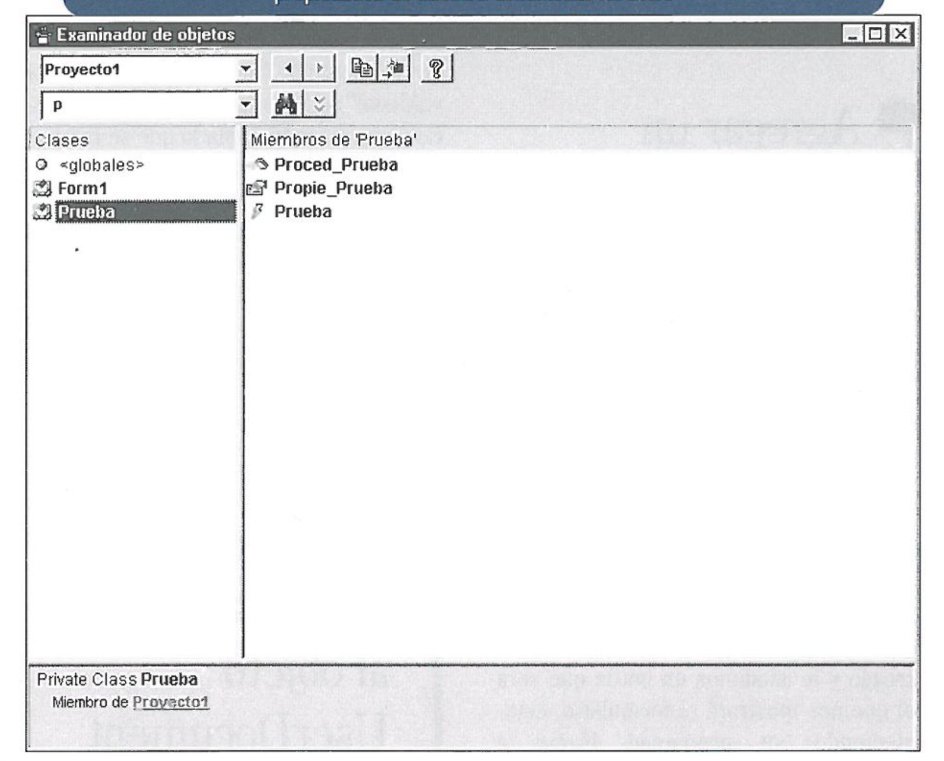


Figura 4. Aspecto del examinador de objetos, a trávés del cual podemos mirar las propiedades de nuestro documento ActiveX.



Para agregar una propiedad, empezaremos agregando un módulo de código (Proyecto + Agregar Módulo). Cambiamos el nombre del módulo a General (hemos de modificar la propiedad Name).

Para ver las propiedades de nuestro documento ActiveX utilizamos el examinador de objetos

Seguidamente en el módulo, en la sección de declaraciones, añadimos el código siguiente:

```
Option Explicit
Public gDActX As DactX
```

Añadimos ahora al documento DactX una segunda caja de texto, estableciendo su valor Text a "" y su valor Name a "txtDActX"

Vamos ahora al código del documento dActX, y seguidamente agregamos un procedimiento (menú Herramientas + Agregar procedimiento) y añadimos una propiedad que se llamará strdActXProp. Modificaremos el código tal y como sigue:

```
Public Property Get strdActXProp() As String
 strActXProp = txtDActX.Text
End Property
```

Y:

```
Public Property Let strdActXProp(ByVal
        vNewValue As String)
 txtDActX.Text = strdActXProp
End Property
```

PropertyBag

Si el usuario final de un documento ActiveX está viendo el documento, puede ser que quiera introducir cierta información, por ejemplo dentro de cuadros de texto, si posteriormente vuelve a leer el documento Active tendrá que volver a hacer dicha tarea otra vez con la consecuente pérdida de tiempo. Podemos ahorrar al usuario un poco de tiempo si ponemos los datos en la propiedad PropertyBag.

La propiedad PropertyBag permite guardar la información que introduce el usuario

Para ello utilizaremos una TextBox, y en su evento Change, colocaremos la instrucción PropertyChanged. Esto hará que cuando el contenido de la TextBox cambie, Internet Explorer pida al usuario que guarde los cambios. Suponiendo que el usuario responda afirmativamente, el código escribirá el valor de la propiedad en PropertyBag. El código que debemos añadir a la caja de texto es el siguiente:

```
Private Sub txtURL_Change()
  PropertyChanged
End Sub
```

Añadiremos el siguiente código al evento WriteProperties del objeto UserDocument:

```
Private Sub
UserDocument_WriteProperties(PropBag
As PropertyBag)
  PropBag.WriteProperty "Prop1", txtURL,
"Hola"
  Debug.Print "Propiedad Escrita"
End Sub
```

Y para recuperar el valor, añadiríamos el siguiente código a la propiedad ReadProperties:

```
Private Sub
UserDocument_ReadProperties(PropBag As
PropertyBag)
TxtURL.text = PropBag.ReadProperty ("Prop1",
"Hola")
  Debug.Print "Propiedad Leida"
End Sub
```

Como agregar Menús a ActXDoc

También podemos agregar un menú a un documento ActiveX mediante el editor de menús, aunque deberemos tener en cuenta que un documento ActiveX no puede tener un menú por si mismo, y por dicha ello el menú que nosotros creamos se combinará con la aplicación que hemos utilizado para ver el documento ActiveX, por lo que deberemos tener en cuenta la negociación de menús cuando agreguemos un determinado menú a un documento ActiveX. Para añadir un menú simplemente vamos al diseñador inicial, ya que es en él donde añadiremos la opción que nos interese (y lo haremos mediante el editor de menús). A través de la opción NegotiateMenu escogeremos el mostrar o no nuestro menú y dónde lo mostraremos, dicha propiedad no está disponible en tiempo de ejecución.

La opción "None (0)" es el valor predeterminado y hace que no se muestre el menú. En la opción "Left (1)" el menú se muestra en el borde izquierdo de la barra de menús cuando el objeto está activo, la opción "Middle (2) hará que se muestre en medio y la opción "Right (3)" que se muestre en el borde derecho de la barra de menús.

Ciclo de vida y eventos de los documentos ActiveX

La vida de un documento ActiveX difiere del ciclo normal de vida de un formulario normal y corriente. Los eventos más importantes a considerar en el ciclo de vida de un documento ActiveX son el Initialize, InitProperties, Show, Hide y Terminate.

El primer evento que se ejecutaría en nuestro documento sería el evento Initialize, seguido de InitProperties y del evento Show. Cada vez que nuestro documento ActiveX se visualice se producirá el evento Show, y mientras exista una referencia a nuestro documento ActiveX en la memoria caché de Internet Explorer (para nosotros, nuestro contenedor), dicho evento se producirá cada vez que se explore el documento ActiveX desde otra dirección URL. Cuando las referencias ya no estén activas se producirán los eventos Hide primeramente y Terminate en segundo lugar.

El evento Initialize, tal y como hemos visto se produce siempre que se carga el documento ActiveX, mientras que el evento InitProperties se producirá cada vez que se guarde el documento. Por lo tanto si lo que deseamos es que cada vez que un determinado procedimiento se ejecute se inicializa el documento, debemos colocarlo en el evento Initialize, pero si solo deseamos que un determinado procedimiento se ejecute cuando el usuario vea por primera vez el documento ActiveX, entonces lo pondremos en el evento InitProperties.

El evento Initialize se produce siempre que se carga el documento ActiveX

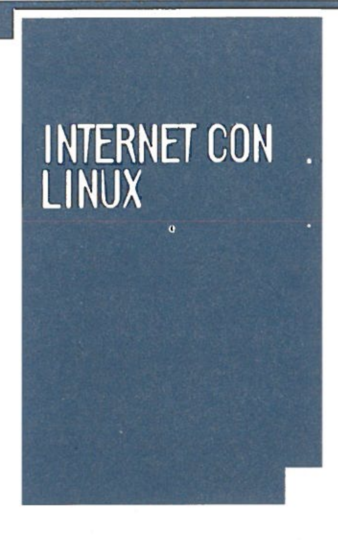
Otro evento muy importante, que hay que conocer, del objeto UserDocument, es el evento EnterFocus. Este evento se produce cuando cualquier objeto, incluido el documento ActiveX, recibe el focus. El evento opuesto seria ExitFocus, el cual se produciría cuando ya no tiene el focus ningún objeto del documento ActiveX, incluido el mismo documento. El evento Resize se produciría siempre cada vez que se cambiase el tamaño del contenedor del documento ActiveX.

El evento Scroll se produce cuando el usuario hace clic en la barra o en la región de desplazamiento del contenedor, o en su defecto cuando el usuario arrastra la barra de desplazamiento. En lo que se refiere a las propiedades, el evento WriteProperties se producirá inmediatamente antes que el evento Terminate, cuando se haya alterado el valor de una propiedad como mínimo (si quisiéramos notificar el cambio al contenedor podríamos utilizar la expresión PropertyChanged).

Scroll se produce cuando el usuario hace clic en la barra o en la región de desplazamiento del contenedor

Cuando el usuario abre un documento ActiveX para el que no se ha guardado ninguna propiedad, entonces los eventos que se producirán son: Initialize (se ha creado el documento pero aún no se ha ubicado o colocado en el contenedor), InitProperties (se han establecido los valores predeterminados de todas y cada una de las propiedades), Show (el documento se muestra en el contenedor y sus propiedades ya están disponibles) y EnterFocus (el documento obtiene el focus). Si el usuario ha guardado alguna propiedad el orden es el mismo pero el evento InitProperties es sustituido por el evento que tiene el nombre de ReadProperties.

Hasta ahora hemos visto algunas generalidades sobre los documentos del tipo ActiveX, muy utilizados en Internet, así como un primer ejemplo. En nuestro próximo artículo podremos ver como ampliar las funcionalidades de los documentos del tipo ActiveX, así como diferentes ejemplos con otros tipos de contenedores tales como el entorno IDE de Visual Basic o el popular cuaderno de Microsoft Office.



El servicio FTP

Carlos Alvaro González
sympha@hotmail.com

Tradicionalmente, el servicio de transferencia de ficheros FTP ha sido el que mayor volumen de información ha movido a través de la Internet a lo largo de su historia.

Desde hace pocos años, el servicio FTP ha sido desplazado en el ranking por el servicio de transferencia de hipertexto HTTP. Sin embargo, el File Transfer Protocol, sigue siendo fundamental, ya que es la manera más limpia y eficiente de enviar y recibir ficheros entre máquinas.

Cliente, servicio, servidor y protocolo

Al igual que hicimos en el caso del hipertexto, veamos estos cuatro conceptos; cliente, servicio, servidor y protocolo, para las operaciones relacionadas con la transmisión de ficheros.

El cliente FTP es un simple programa que nos introduce en un shell; esto es, un programa capaz de reconocer comandos, ejecutarlos y devolvernos respuestas, todo ello de manera interactiva. Es el "file transfer program", y suele llamarse "ftp.exe" en entornos MS-DOS y OS/2, y "ftp" en entornos UNIX y otros.

El servicio que proporciona la realización del intercambio de ficheros entre la máquina servidor y la cliente se denomina servicio de FTP, y requiere la presencia de un proceso daemon, habitualmente llamado File Transfer Protocol Daemon (ftpd) que, manteniéndose a la escucha del socket 21 en el sistema del servidor, recibe y atiende las peticiones de los programas cliente. Es, por tanto, ftpd el que juega el papel de servidor en la transferencia.

El protocolo TCP/IP encargado, a nivel de aplicación, de servir o recoger los ficheros es File Transfer Protocol (FTP), implementado en el cliente ftp y en el servidor ftpd.

Al igual que ocurría con el servicio de transmisión de hipertexto, el servidor ftpd puede ser invocado por el superserver inetd, o correr sobre la máquina independientemente.

Manejo del cliente ftp. Anonymous ftp

El protocolo de transferencia de ficheros (FTP) permite a un usuario de una computadora recibir ficheros de otra computadora o enviarlos. La seguridad se maneja mediante la petición de un nombre de usuario y una password por parte del sistema remoto. Deben estar contempladas las características propias de las diferen-

tes máquinas, tales como el uso de juegos de caracteres diferentes o convenciones para señalar el fin de línea.

Como ya sabemos, el File Transfer Protocol permite transferir ficheros entre dos ordenadores, generalmente conectados a la Internet. La mayoría de los comandos de FTP están orientados al tráfico de ficheros. Algunos son iguales o de sintaxis similar en diferentes máquinas, pero otros no. Usualmente, FTP listará los comandos disponibles tecleando "help" o "?". También se puede obtener ayuda desde utilidades del sistema tales como "man" o "help", tecleando por ejemplo: "man ftp" o "man ftpd".

Algunos comandos importantes, disponibles en la mayoría de los sistemas, son:

get	Copia un fichero desde el ordenador remoto al local.
put	Copia un fichero desde el ordenador local al remoto.
ls / dir	Lista los ficheros del directorio actual.
cd	Cambia de directorio.
pwd	Visualiza el directorio actual.
binary	Establece el modo binario de transferencia.
ascii	Establece el modo ASCII de transferencia. (Por defecto).
open	Intenta abrir conexión contra una máquina remota.
user	Especifica el usuario mediante el que intentamos ganar el acceso.
close	Cierra la conexión en curso.
bye /quit	Abandona el interface FTP.

El modo de transferencia (ASCII / binary) sirve para habilitar o deshabilitar la conversión de ficheros. Por ejemplo, consideremos la petición de un fichero, por parte de un cliente Unix a un servidor OS/2. OS/2 (al igual que MS-DOS) representa el EOL (fin de línea) mediante los caracteres ASCII 13 y 10, mientras que Unix sólo utiliza el 10. Un fichero de texto que pasa de un sistema OS/2 a un sistema Unix debe sustituir los pares [13][10] por un único caracter [10]. Esto implica modo de transferencia ASCII. Sin embargo, si el fichero que transferimos no es de texto, cualquier par [13][10] no tiene ese significado de "fin de línea", y no debe producirse el cambio. Es el modo de transferencia binary.

Obviamente, las modificaciones a los ficheros se hacen en local trabajando con una copia. Muchos sistemas ofrecen, a través de la Internet, ficheros a Anonymous FTP. Estos incluyen software, documentos de varias clases, ficheros de configuración de redes, archivos de correo electrónico, listas, etc.

El cliente FTP es un simple programa capaz de reconocer comandos, ejecutarlos y devolvernos respuestas de manera interactiva.

Anonymous FTP es una facilidad ofrecida por muchas máquinas de la Internet, que permiten conectarse con el nombre de "anonymous" o de "ftp" y como password la dirección e-mail. (Esto último no es necesario pero se hace como cortesía). La password también puede estar establecida como "guest" o una palabra similar. Se permite el "movimiento" por un árbol de directorios y recoger ficheros, aunque, habitualmente, no se permite almacenar los propios. Es típica la existencia de un directorio "pub" donde se pueden encontrar cosas interesantes, así como la un fichero "ls-lR" que contiene una lista completa de los ficheros disponibles.

Normalmente, los ficheros se encuentran empaquetados en "archive files", que pueden estar o no comprimidos. El formato más común para los archive files es tar. Ocasionalmente se usan ficheros shar en lugar de ficheros tar. Los ficheros shar son shell archives, que se desempaquetan con el comando unshar, o ejecutándolos directamente desde el Bourne Shell.

Para ficheros comprimidos, la extensión .Z es indicativo del programa compress, aunque también se utilizan otros, como zoo y arc, que combinan formatos de almacenamiento y compresión.

Algunos tipos comunes de fichero son:

.tar	Tape ARchive
.shar	SHell ARchive
.Z	Compress / uncompress
.z	Pack / unpack
.arc	Arc
.zoo	Zoo
.gz	Gzip
.uu	Uuencode / uudecode

El método más simple para iniciar FTP es tecleando el comando "ftp <host>" ante el prompt del sistema, donde "host" es el nombre del sistema remoto o su dirección IP. Una vez que la conexión se haya establecido satisfactoriamente, se pedirá un nombre de usuario y una password. Es aquí donde se puede utilizar Anonymous FTP, con lo que se obtendrá un acceso restringido al sistema.

Finalmente, aparece el prompt de FTP, usualmente "ftp>".

Configuración del servidor ftpd

El servicio de transferencia de ficheros utiliza, como es habitual en Unix, una serie de ficheros para configurar el funcionamiento del servidor ftpd.

La autentificación, a la hora de permitir o denegar el acceso, es sencilla pero efectiva. Lo normal es que el permiso dependa de estas dos únicas reglas de autentificación:

1. El username debe estar en /etc/passwd. Si la password es nula, se le requerirá al usuario antes de cualquier operación con ficheros.

2. El username no debe aparecer en /etc/ftpusers y el shell del usuario debe aparecer en /etc/shells.

Si embargo, si nuestro sistema va a permitir el acceso para anonymous ftp, deberemos extremar la seguridad, ya que estamos abriendo las puertas del sistema, no sólo a nuestros usuarios registrados, sino a cualquiera que disponga de medios para alcanzar nuestra máquina a través de una red. En este caso debemos tener en cuenta lo siguiente:

1. Debe existir un usuario 'ftp' en /etc/passwd. Las operaciones realizadas por el daemon del protocolo correrán bajo el perfil y los permisos de este usuario, por lo tanto, cuanto más bajo sea su nivel de autorización, más seguros serán los accesos. Su ID (identificador de usuario) debe ser el de un usuario no privilegiado. Su HOME (directorio de conexión) será, visto desde fuera, la raíz del sistema, es decir, cuando "anonymous" accede al sistema, lo único que verá será lo que haya por debajo del valor de HOME. Este cambio de directorio raíz para el usuario "ftp" se realiza mediante la llamada al sistema 'chroot'.

2. La password encriptada de 'ftp' debe ser '*' para evitar telnet. El algoritmo de encriptación de passwords de Unix garantiza que no existe ninguna combinación de caracteres de la tabla ASCII cuyo resultado sea '*'.

3. La restricción de /etc/ftpusers es válida. En un momento determinado, podremos denegar el servicio a Anonymous FTP, incluyendo la línea 'ftp' en este fichero.

4. La restricción de /etc/shells no es válida.

5. La construcción del directorio de Anonymous FTP, que en nuestra distribución de Linux es, por defecto, "/home/ftp", debe realizar se teniendo en cuenta las siguientes reglas:

   a. /home/ftp debe ser propiedad del usuario ftp y con acceso de escritura prohibido para el resto de los usuario.

   b. /home/ftp/bin debe ser propiedad de root, sin acceso de escritura al resto de usuarios y debe contener el comando "ls", en modo 111 (sólo permiso de ejecución) para dar soporte a los comandos de list (habitualmente, "dir" y "ls") del servidor ftpd.

   c. /home/ftp/etc debe ser propiedad de root, sin acceso de escritura al resto de usuarios y debe contener los ficheros "passwd" y "group", con objeto de poder hacer la tradución de userid a username. El contenido de esos ficheros deben ser las líneas indispensables de los originales /etc/passwd y /etc/group, con objeto de dar la menor información posible sobre los usuarios y directorios del sistema a potenciales hackers.

   d. /home/ftp/pub debe ser propiedad de ftp, sin acceso de escritura al resto de usuarios. Es el directorio público de ficheros que se ofrecen a través de la red. No es una buena política el permitir la escritura en el mismo. Los usuarios registrados del sistema, que pueden estar autorizados a escribir ficheros para intercambiar entre sí, lo pueden hacer en directorios invisibles para Anonymous FTP.

El fichero .netrc

El fichero .netrc permite automatizar accesos FTP a hosts. Es muy útil cuando tenemos procesos que se realizan periódicamente, siguiendo un patrón, y siempre de la misma forma. Reside en el directorio de conexión (HOME) del usuario y está compuesto por una serie de elementos, separados por IFS (Internal Field Separator; habitualmente, espacio en blanco, tabulador o newline).

Supongamos el caso de la actualización de versiones: tenemos una máquina, llamada "andora", donde periódicamente se genera un fichero "/home/ftp/pub/datos" que debemos obtener vía ftp para procesarlo en local.

El fichero .netrc se puede utilizar para automatizar los accesos FTP a hosts

El usuario "charlie" podría tener un .netrc, en su directorio de conexión, como el siguiente:

```
machine andora login anonymous password
   charlie@menteuse.intranat init
get /pub/datos
bye
```

Con esto, cada vez que, desde menteuse el usuario "charlie" teclee "ftp andora" se conectará automáticamente como Anonymous FTP, traerá el fichero "/home/ftp/pub/datos" y cerrará la conexión.

Conclusión

Con nuestro servicio de transferencia de ficheros activo y disponible, deberemos hacer un meticuloso seguimiento de la ocupación de nuestros discos duros, ya que es el servicio por excelencia de transferencia masiva de información.

Al igual que decíamos en el artículo anterior, hablando del servicio de hipertexto, con el de transferencia de ficheros también se precisa de una persona encargada de su gestión y mantenimiento. La actualización del software, que ofrecemos a través de este servicio y la inclusión del nuevo, debe ser vigilada para ofrecer una calidad a los clientes.



Máquinas recreativas basadas en Intel

Jorge Figueroa
erde@arrakis.es

La plataforma Intel ha evolucionado tanto, en lo que a rendimiento de procesado se refiere, que actualmente ya es apta para ser usada en la creación de máquinas recreativas de salón. Estas máquinas se están diseñando con un estándar llamado O.A.A.

Estas máquinas tendrán la ventaja de ser un estándar abierto y los juegos que se produzcan para las mismas serán convertibles a versión doméstica PC muy fácilmente, sin apenas tener que realizar cambios. A continuación, se detalla todo lo referente a este nuevo estándar llamado OAA.

Las máquinas basadas en arquitectura Intel, han aumentado mucho en los tres últimos años la potencia de cálculo y gráfica, gracias principalmente a los nuevos Pentium II, que incorporan una unidad de coma flotante de alto rendimiento, a las extensiones MMX y a la tecnología de bus AGP, que ahora se empieza a implementar para los gráficos 3D en las nuevas placas gráficas.

Con todo ello, ahora se han encontrado que dicha plataforma se ha convertido en apta para sectores que antes le estaban vedados, como es por ejemplo el de las máquinas recreativas, negocio que mueve anualmente la cantidad de 16 billones de dólares en todo el mundo, y que sólo en los Estados Unidos dispone de un parque de 1,2 millones de máquinas recreativas instaladas.

Visto esto, Intel decidió unirse con otras empresas importantes de este sector y crearon el llamado estándar O.A.A., que son las siglas provenientes del inglés <*Open Arcade Arquitecture*> (Arquitectura abierta para máquinas recreativas) y que son un conjunto de directivas hardware, que conforman el "white paper" para la creación de máquinas recreativas basadas en arquitectura Intel.

En este artículo se va a detallar todo lo referente a este estándar, para que las posibles empresas y grupos de programación desarrolladores de soft de entretenimiento, que puedan estar interesados en el desarrollo de videojuegos para recreativas, sepan como está la situación actualmente.

Historia de la O.A.A.

En 1996, Amusement , la Music Operator Association (AMOA) y otros desarrolladores de PC's, se reunieron para intentar llevar la plataforma PC a la industria de máquinas recreativas y en Diciembre de ese mismo año, Intel se reunió con más de 160 líderes desarrolladores de recreativas para realizar una demostración de la capacidad de su arquitectura para ser usada como plataforma base de videojuegos de salón.

El 24 de Marzo de 1997, Intel anunció la iniciativa VCI (<Visual Computing Initiative), según la cual, tenían como objetivo aumentar el rendimiento gráfico de los PC's en 10 veces en un plazo de 3 años.

Actualmente la especificación O.A.A. va por la versión 0.6, y en Mayo de este año, se han presentado las primeras máquinas hardware para el desarrollo de videojuegos basados en esta arquitectura. (Máquinas de testeo de juegos para desarrolladores).

En pocos años los procesadores Intel han aumentado mucho sus prestaciones

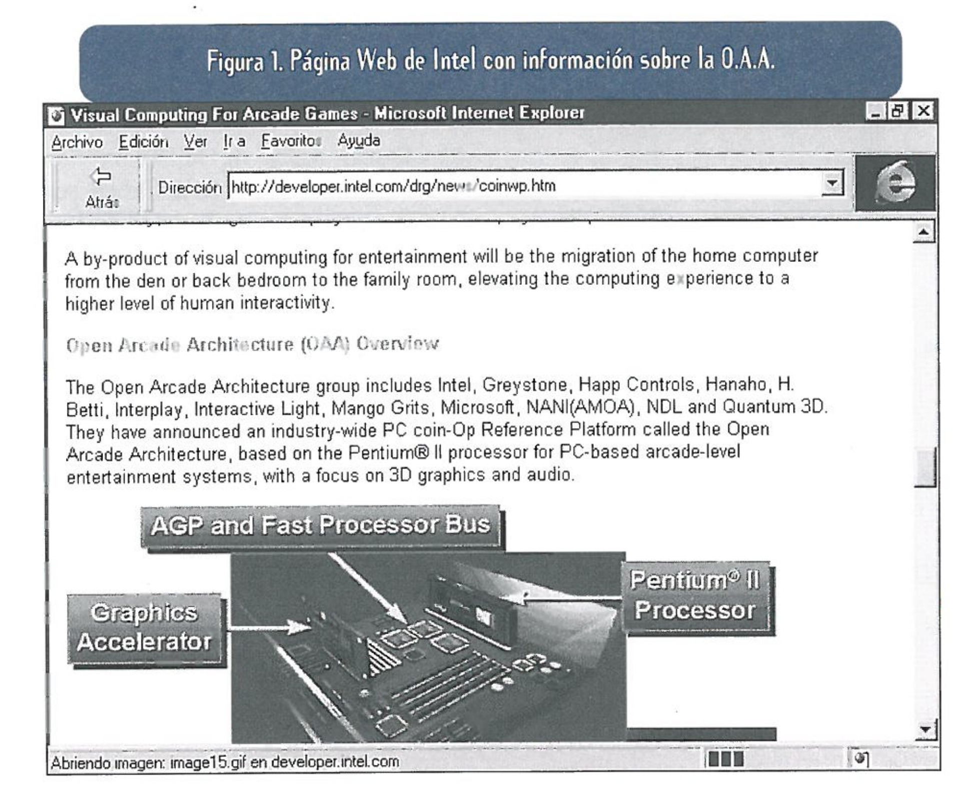
Durante este año, también se han definido las extensiones para el desarrollo de máquinas multiprocesador y a partir de principios del año que viene, empezarán a aparecer en todo el mundo los primeros videojuegos de diversas marcas basados en este nueva plataforma hard.

Objetivos de la OAA

Este estándar persigue tres principios claros:

1º. El principal objetivo de utilizar recreativas basadas en esta arquitectura, es que se podrán convertir las versiones originales de los videojuegos de recreativa a juegos de PC doméstico sin casi tener que retocar el programa, ya que a nivel de código será igual y tan sólo se necesitará reducir un poco las prestaciones y requerimientos de los mismos para que sean adaptables a las configuraciones de sobremesa, con el consecuente ahorro de tiempo y personal que significa todo esto, lo que se traduce a su vez en ahorro económico y en un mayor mercado disponible, el de salón y el doméstico.

Figura 1. Página Web de Intel con información sobre la O.A.A.



2º. Las máquinas realizadas según este estándar pueden albergar cualquier juego, ya que son plataformas genéricas (como dice el título, una plataforma abierta), por lo cual no será necesario crear hard específico para cada máquina o videojuego, con el consiguiente ahorro de dinero por parte de los diseñadores.

Actualmente la arquitectura Intel es apta para crear máquinas recreativas

3º. Junto con este estándar, se pone a disposición de los diseñadores de videojuegos también las tecnologías de red local y red Internet, con lo que se facilita la creación de videojuegos mutijugador en tiempo real.

4º. Estas máquinas serán fácilmente ampliables (como los ordenadores normales) y se ahorrará mucho dinero en actualizaciones y ampliaciones.

Componentes de la O.A.A.

La Asociación O.A.A. incluye a muchas empresas como Intel, Greystone, Happs Controls, Hanano, Hi Betti, Interplay, Iinteractive Light, Mango Grits, Microsoft, NANI (AMOA) y Quantum 3D y los apartados a los que se dedican cada una de ellas son más o menos los siguientes:

- Hanano está dedicada a la creación de cabinas arcade y los sistemas de integración.
- Happs Controls desarrolla controladores y sistemas de integración.
- Intel aporta los procesadores (Intel Pentium II), las placas base y hardware gráfico.

- Microsoft ha aportado su sistema operativo, el software del sistema y diferentes herramientas de desarrollo.
- Quantum 3D y 3Dfx Interactive desarrollan placas gráficas 3D de alto rendimiento y Metatools cámaras 3D.
- Breton Pacific (Hi Betti) se dedica a la distribución de las máquinas recreativas (entre otros).

Intel ha llamado O.A.A. al "white paper" de las recreativas basadas en su arquitectura

Para crear una máquina recreativa que cumpla con las normas indicadas en la O.A.A., la plataforma debe cumplir una serie de requisitos hardware y software.

Requisitos de la especificación OAA

Los requerimientos de una plataforma basada en este estándar, son más o menos que posea como mínimo un procesador Pentium II con 512 KB de caché level 2 funcionando a al menos a 266 Mhz de velocidad de reloj, debe estar instalado en una placa preparada para poder funcionar a velocidades mayores a la del procesador que esté inicialmente instalado. Esta placa debe disponer de al menos 3 slots PCI 2.1 y un mínimo de 2 slots ISA y también debe disponer inicialmente de un mínimo de 32 Mbytes de memoria RAM instalados, los cuales deben poderse ampliar como mínimo hasta los 256 (puesto que no sabemos lo que requerirá la aplicación que instalemos en ella).

En lo que se refiere a la BIOS de la placa base, debe tener unos requerimientos mínimos que son estos:

1º Poseer salida de televisión (los estándar NTSC y PAL deben ser cambiables por medio del SETUP).

2º Los puertos USB deben poder activarse/desactivarse también por medio del SETUP.

3º Debe soportar el National 308 GPIO.

4º Debe tener opciones para poder autodetectar el tipo de disco duro IDE instalado.

5º Al ser un sistema accesible por muchas personas, debe soportar dos modos de SETUP, el de usuario y el de administrador.

6º La BIOS debe soportar el modo 3 de dispositivos de disquetes para poder ser compatible con el estándar japonés y sin lo cual la máquina no sería exportable a dicho país.

7º Debe soportar la especificación PLUG&PLAY v.1.0a, que está soportada a su vez por el sistema operativo Windows 95.

8º debe poder seleccionarse el lenguaje que usa la SETUP entre las principales lenguas mundiales, que son: Japonés, Francés, Italiano, Alemán, Español e Inglés.

9º El controlador IDE debe tener capacidades ATA-33 para poder usar discos duros de alta capacidad, incluyendo el modo LBA y los modos avanzados PIO 3 y 4 para poder acceder a una mayor velocidad a los discos PCI- IDE, los cuales serán autoconfigurados automáticamente por la BIOS.

10º Debe haber 4 KB de espacio libre reservado para poder almacenar un Logotipo propio que aparecerá durante el arranque.

11º La BIOS debe ser actualizable, basándose para ello en memoria FLASH reescribible.

12º Debe soportar de arranque ya CD-ROM's mediante la especificación conocida como "El Torito".

13º La BIOS debe soportar el estándar DPMI 2.0 de memoria.

La controladora IDE instalada debe tener capacidad para soportar hasta 4 dispositivos conectados simultáneamente (EIDE), también debe soportar el método de transferencia Ultra DMA 33 síncrona y, como opciones, puede disponer de un reproductor CD-ROM con una velocidad de 16x y/o un lector de DVD.

El sistema debe tener instalado un mínimo de 4 MB de memoria de vídeo de placa gráfica, la cual puede ser por ejemplo una Virge S3, una ATI 3D Rage Pro o una Matrox Mystique u otra similar. Es muy importante que soporte el direct draw 5.0 o superior y Windows 95/NT. Además de todo esto, debería soportar resoluciones de hasta 1024*768 con 16 bits de profundidad. El refresco de pantalla debe ser de al menos 60 Hz y la memoria de vídeo del tipo SGRAM o SDRAM para una mayor velocidad de acceso.

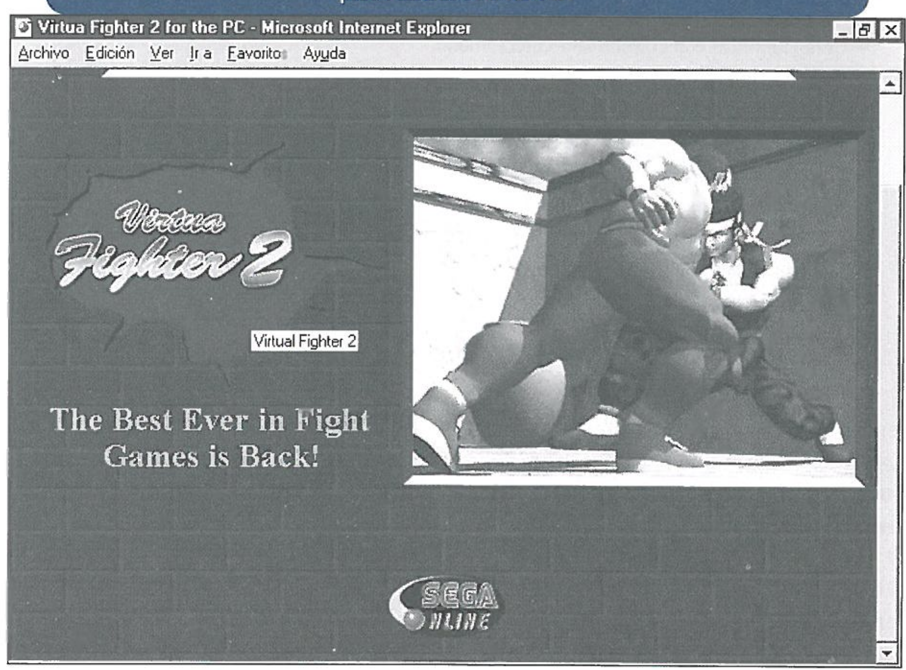
Estas máquinas llevaran aceleradoras 3D basadas en los nuevos buses AGP

La máquina debe poseer instalada una placa aceleradora 3D que de un rendimiento de 400 Kpolígonos por segundo y que pueda dar un procesado sostenido de 45 Megapíxels por segundo con cálculos de filtrado bilineal, Sombreados Gouraud, Z-buffer y con color de 16 bits de profundidad. Para las texturas, la placa debe disponer de al menos 4 Megabytes de memoria exclusivos para las mismas. Si puede ser, la placa aceleradora 3D debe

ser un modelo que use una conexión de puerto AGP de 64 bits de doble o cuádruple velocidad (x2 o x4). Algunos modelos ya disponibles son: ATI 3D Rage Pro, Cirrus Laguna3D, Trident 3Dimage975 y la NVIDIA's RIVA1289.

En el apartado de la salida de televisión, el sistema debe disponer de una salida de televisión NTSC (PAL S-Video) para poder conectarlo a televisores grandes. Este dispositivo, debería disponer de selección de señal, posicionamiento de imagen y que además todo ello fuese controlable mediante el propio software y la BIOS del sistema. Un decodificador MPEG-2 HW, el cual será necesario en aplicaciones que estén basadas en discos DVD. Deberá cumplir los estándar MPEG-1 y MPEG-2 y soportar una tasa de datos de 15MB/s a 720*480 con ratios de aspecto 4:3 y 16:9 a 30 imágenes por segundo.

Figura 2. Juegos como Virtua Fighter 2, demuestran la validez de la arquitectura Intel para realizar recreativas.



Los juegos podrán usar redes locales o Internet para crear juegos multijugador

El apartado de la capacidad de poder capturar vídeo es sólo posible bajo ciertas condiciones de luz ambientales y no es muy recomendable de usar porque consume mucho tiempo de procesado, lo cual repercute en la propia aplicación, que dispondrá entonces de recursos muy limitados para la generación de gráficos y demás cálculos.

Como es evidente, el sistema deberá incorporar un sistema de sonido potente, el cual debe ser compatible con el *Microsoft 3D Direct Sound* y con los estándar mundiales Sound Blaster Pro, Adlib y Roland MPU-401. Debe poseer capacidad para reproducir sonido a 44 Khz a 16 bits (calidad CD) y disponer de un decodificador (ya sea software o hardware) AC-3 para la generación de sonido Dolby Surround. Opcionalmente, se recomienda que posea un sistema de música por síntesis de tabla de Ondas que permita ampliación por hardware y que cumpla con la especificación estándar AUDIO'98. La cabina de juego, debería disponer como mínimo de altavoces de alta calidad estéreo o sonido cuadrofónico y que en caso de no estar disponibles, se deberá haber un jack de audio (un conector) para posibles jugadores que quieran disfrutar del sonido con cascos. En la tabla 1 puede ver un listado de todos los requerimientos de audio de la plataforma.

En lo que a conectividad con el exterior se refiere, la máquina debe disponer de 2 conectores Universal Serial BUS (USB) que cumplan las especificaciones de la versión 1.0 y que soporten velocidades de transferencia de entre 1.5 y 12 MB por segundo. Además, deben poseer el microcontrolador incorporado y ser compatibles con la APM 1.2 y la ACPI. Para crear juegos multijugador, la máquina debe disponer de alguna de las siguientes capacidades: la primera opción es la de poseer de conexión Ethernet a 100 MB por segundo de ancho de banda, como es por ejemplo la Intel EtherExpress PRO/100B PCI LAN o una equivalente. La segunda posibilidad es la de poseer un módem de 56.6 Kbps que sea compatible con el estándar H.324, preferiblemente con línea telefónica del tipo ISDN / T1.

El sistema de alimentación debería disponer de un controlador termoestático, un Corrector de factor de alimentación (PFC), un tbd Watts y un APM para la protección contra cambios de voltaje, que podrían dañar el hard de la máquina. De esta forma queda cubierta la protección ante cualquier posible problema eléctrico.

Las aplicaciones funcionan a resoluciones de 800x600 o 1024x768

El tipo de monitor de salida se deberá elegir según las características particulares de la máquina, según su diseño y tipo de juego que vaya a incorporar. Hay dos configuraciones básicas posibles: una opción es la de instalar un monitor SVGA de 21 pulgadas con una resolución de 800*600 o 1024*768 con refresco de

Tabla 1. Requerimientos de sonido.

- Compatibilidad con el Microsoft Direct Sound.
- Que sea capaz de conseguir la certificación Dolby CLass "B".
- Que sea compatible SoundBlaster en ventana DOS.
- Sintetizador FM con un mínimo de 4 voces y 22 operadores.
- Síntesis de ondas descargable por hardware.
- Soportar reproducción de sonido digital a 11,22,32,44.1 y 48 Khz.
- Soportar modo Full Dúplex.
- Capacidad de posicionamiento 3D.
- Que cumpla la especificación APM 1.2.
- Conector MIDI/Juegos.

entre 60 y 75 H, los cuales resultan muy caros en caso de que sea para jugar con juegos típicos, o instalar 3 monitores, que es preferible cuando sean juegos de vuelo y simuladores 3D de inmersión real (1 vista frontal y las dos laterales), aunque entonces se requieren varias placas gráficas con soporte de varios monitores. La segunda opción es la de instalar un monitor grande de alta resolución que seguramente poseerá una definición de 640*580 o 768*480 con un tamaño de 35 pulgadas o más. Esta opción es más adecuada en aplicaciones donde el jugador se sienta para jugar a cierta distancia de la pantalla, como son los juegos de carreras con simulador de asiento y carrocería, puesto que para juegos donde nos situemos a corta distancia de la pantalla , perderemos calidad de imagen si usamos esta opción.

Todas las máquinas funcionarán bajo Windows 95 o NT

También es importante que se puedan instalar periféricos típicos como teclado, conectable mediante PS/2 o USB. Ratón, mediante conector PS/2, en el puerto serie o el USB. Como complemento, se pueden añadir slots ISA extras para poder conectar dispositivos de entrada no-estándar.

La máquina debe poseer un puerto estándar RS-232 y el puerto serie (COM), puesto que muchas compañías fabrican periféricos basados en estos conectores, como son Joysticks de diversos tipos, botones, trackballs, volantes, pedales, dispositivos de vuelo, etc. Adicionalmente, el PC debe poder conectarse con dispositivos de pago de monedas y lectores de tarjetas de crédito para posibles sistemas de pago mediante tarjeta.

En el apartado de seguridad, las máquinas recreativas deben estar protegidas contra los jugadores para que estos no puedan acceder a los botones de reseteo o a unidades estándar de entrada mediante los cuales puedan introducir virus en el sistema o alterar su funcionamiento. Como máximo, la máquina deberá disponer de un teclado numérico en aquellos casos que el sistema incluya posibilidad de pago con tarjeta, al estilo de como funcionan los cajeros automáticos, para que el jugador pueda introducir los datos cómodamente. En los casos de que los juegos posean capacidad multijugador, los oponentes deben poder seleccionarse mediante el propio joystick o sistema de control que esté instalado para jugar o mediante la pulsación sobre la pantalla táctil si es que esta capacidad está disponible.

En lo que a normas de incitación al juego se refiere, la máquina sólo puede usar como factor clave de atracción para el publico lo que se muestra en la pantalla cuando no hay nadie jugando, puesto que es lo único mediante lo cual se puede incitar al mismo a que gaste dinero. Lo normal es hacer aparecer secuencias atractivas de demostraciones del juego en acción y con la posibilidad de mezclarlo con secuencias de vídeo digital que sirvan para introducir el argumento del mismo.

En http://www.intel.com, está toda la información disponible sobre el tema del artículo

Para finalizar, sólo decir que todo el software que se desarrolle, se debe crear para Windows 95 o Windows NT, aunque se debe tener presente que si se opta por la versión NT, no podrá usar las librerías DIRECT X puesto que aún no están disponibles para este sistema operativo, lo cual puede ser un inconveniente más si tenemos en cuenta que estamos hablando de videojuegos. Otro dato importante es que el sistema operativo instalado en el sistema en la plataforma de destino, estará configurado en arranque en modo de alta protección, con lo que muchos de los recursos del sistema no estarán disponibles por seguridad y lo cual deberán tener en cuenta nuestras aplicaciones. El GUI del sistema operativo , sólo estará accesible para el personal de mantenimiento el cual se supone dispondrá de la clave de acceso al sistema.

Conclusión

Con este estándar, se ha abierto las puertas del sector de las recreativas para que muchas más empresas desarrolladoras de soft puedan crear a partir de un futuro más o menos inmediato juegos para máquinas de salón de forma fácil y barata sin tener que adaptarse a arquitecturas cerradas y sin necesidad de licencias especiales.

Si se desea ampliar la información sobre la OAA, puede acceder a la Web de la propia Intel **<www.intel.com>**.

SoftFactory/2000

J.Carlos Gómez

Aunque últimamente ha dejado de ser un tema de actualidad en los informativos, el año 2000 sigue siendo una fuente de problemas para los grandes sistemas informáticos y los días van pasando.

Micro Focus ha estudiado el problema del año 2000 durante años y ha sacado al mercado una interesante solución, que ahorrará tiempo y dinero a los departamentos de informática de las grandes empresas, que ven como van pasando los días que quedan para solucionar el problema.

Una aproximación al problema

Los departamentos de informática, que tuviesen constancia del problema del año 2000 y lleven años trabajando en el problema, tendrán ya muchos programas actualizados y no necesitarán de herramientas especiales, pero tal y como ha demostrado un estudio de Micro Focus, la mayoría de las empresas, que utilizan Cobol, no han adoptado todavía ninguna solución y no tienen ni tiempo ni recursos para encontrar una solución rápida y económica al problema.

SoftFactory/2000 es una herramienta que permitirá localizar automáticamente las partes críticas del sistema

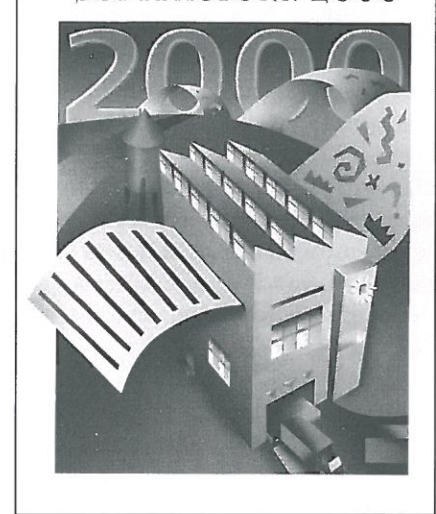
Ante esta problemática, Micro Focus ofrece al mercado de los grandes sistemas una herramienta que permitirá localizar automáticamente las partes críticas del sistema, que deberán ser retocadas posteriormente. Existen varios productos que utilizan la filosofía de buscar y mostrar todas las partes que manejan fechas en un código, pero este tipo de búsquedas provoca enormes listados, que posteriormente han de ser revisados por un gran número de analistas.

La solución al problema

Con SoftFactoy/2000 no se ofrece un simple localizador de fechas en el código, sino un sistema inteligente, que ahorrará tener que examinar un montón de líneas de código innecesarias. Este sistema ahorrará un montón de tiempo y dinero a las empresas. El producto es capaz de detectar sólo los puntos críticos donde el manejo de fechas puede provocar errores en el año 2000, llegando incluso a simular fechas virtuales para comprobar si los cambios realizados posteriormente tendrán los efectos deseados.

La mayoría de sistemas informáticos no están todo lo documentados que debieran, lo que hace que tomar el contacto con toda la realidad de un gran sis-

Figura 0.



tema informático requiera grandes dosis de tiempo y paciencia. Durante las fases de Inventario y Evaluación, un módulo de SoftFactory/2000, llamado Resolve/2000, se encarga de proporcionar a los programadores y analistas una representación gráfica normalizada de cómo funcionan las aplicaciones del sistema. Este herramienta permite, de una forma automática, obtener en unos segundos una completa estructura del sistema, función que será de suma utilidad a los analistas, tanto del proyecto Año 2000, como del resto de aplicaciones del sistema (véase figura 1).

Una vez creada la estructura del sistema en el programa, un segundo módulo, llamado SmartFind (véase figura 2), se encarga de examinar y mostrar la lista de posible fechas e incluso mostrar el código fuente correspondiente en un editor integrado. Esto permite comprobar dónde están los puntos críticos en cada una de las partes del código e incluso realizar modificaciones y comprobar los resultados sin afectar al sistema central, ya que es posible trabajar sobre un sistema de PC´s en red, que emule a los grandes sistemas. Esta técnica de emulación ahorrará los siempre escasos recursos del ordenador central, que no puede dejar bajar sus rendimientos para hacer pruebas y mantendrá a salvo su integridad al trabajarse en un equipo externo que lo emula perfectamente.



MICRO FOCUS

Resolve/2000, se encarga de proporcionar a los programadores y analistas una representación gráfica normalizada de cómo funcionan las aplicaciones del sistema

Otra de las herramientas que agradecerán los jefes de proyecto del Año 2000 será la posibilidad que ofrece el programa de generar informes de evaluación de todo el proyecto, pudiendo estimar los recursos humanos y costes asociados al mismo, basándose en datos reales del sistema y no en aproximaciones que pueden producir errores.

La opinión

Las empresas que necesiten resolver el problema del año 2000 disponen de varias alternativas para afrontarlo con éxito:

- Expansión de campos: la idea es examinar todas las líneas de código e ir cambiando el tamaño destinado al campo que guarda el año, pasándolo de dos a cuatro posiciones. Este proceso es el más lento y salvo que se lleven ya varios años trabajando en él, será muy difícil empezarlo a implementar a sólo menos de tres años de la fecha crítica.

- Técnica de ventanas: la idea es establecer una fecha base, que corresponda al siglo XX y, por convenio, establecer que todas las fechas ante-

Figura 1.

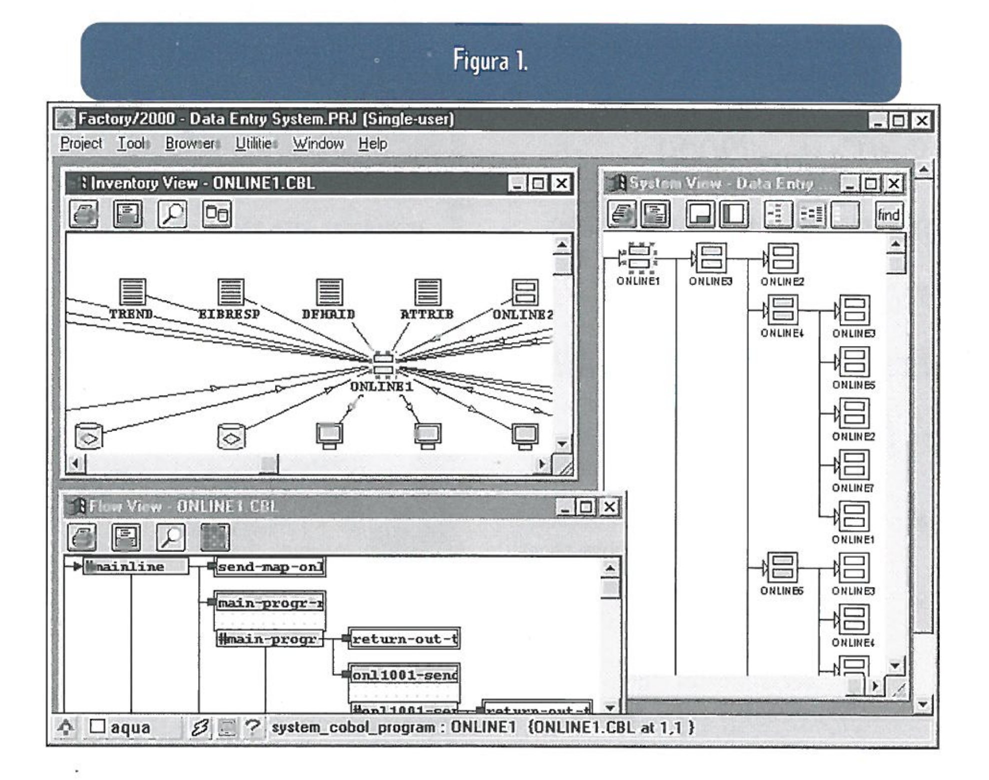
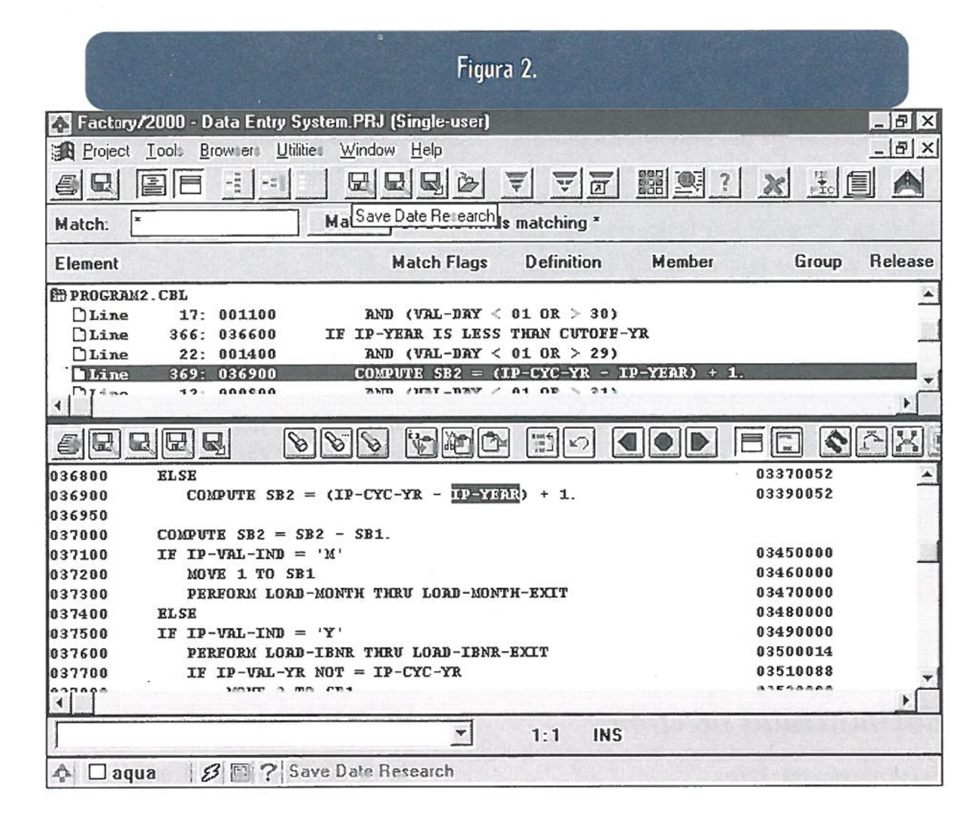


Figura 2.



riores a esa sean del siglo XXI. Por ejemplo, se puede establecer que todas las fechas que tengan dos posiciones y sean del tipo 70 correspondan a 1970 y las anteriores sean del siglo XXI. La gran ventaja de esta técnica es que no obliga a cambiar el tamaño de los campos dedicados a las fechas.

SoftFactory/2000 permiten minimizar los recursos humanos necesarios para resolver el proyecto 2000

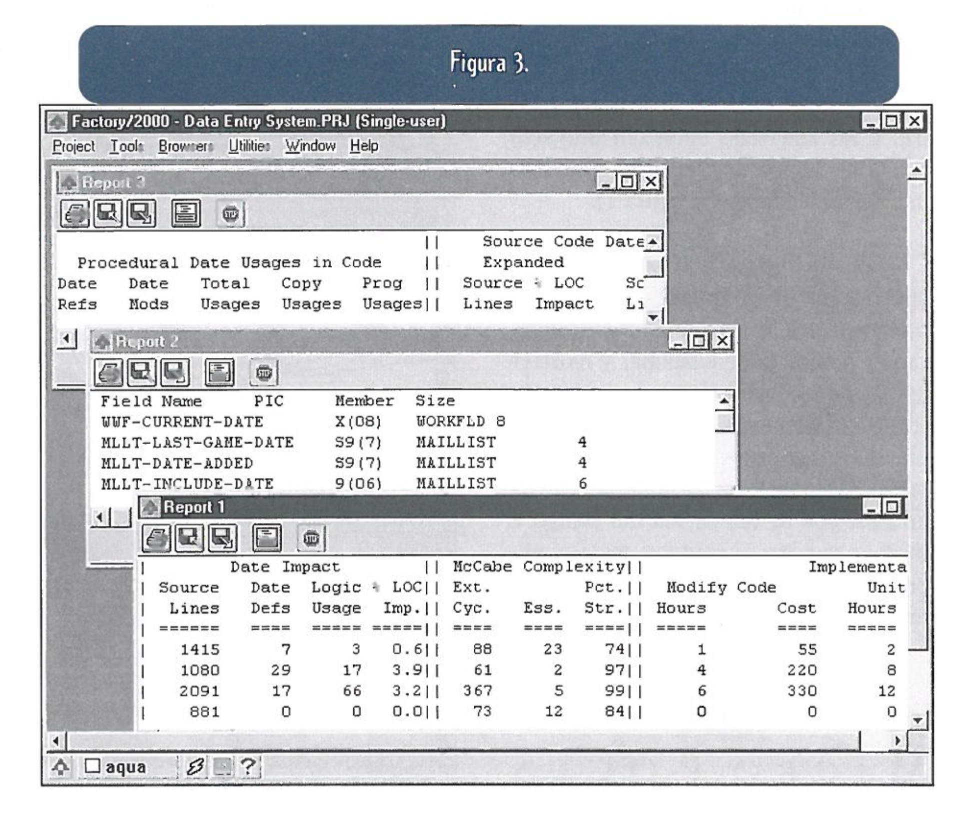
- SoftFactory/2000: esta aplicación utiliza las ventajas de ambas técnicas, obteniendo unos resultados que simplificarán al máximo el problema.

Las herramientas del sistema SoftFactory/2000 permiten minimizar los recursos humanos necesarios, realizar la mayoría de análisis de la aplicación automáticamente (para reducir el número de líneas de código a estudiar) y minimizar el riesgo e impacto sobre el sistema central cuando se realizan pruebas y modificaciones del código. En resumen, SoftFactory/2000 es un producto a tener muy en cuenta por las grandes empresas, que tiemblan al ver como los días van pasando y el Apocalipsis informático se acerca.

Según estudios de Micro Focus, SoftFactory/2000 puede llegar a reducir por tres el coste total de la crisis del año 2000, una buena muestra de esta realidad es que el sistema de Micro Focus permite que los resultados obtenidos en un día de trabajo sean los equivalentes a unos veinte días de trabajo con otros sistemas de la competencia.

El director general de Micro Focus Year 2000 Solution está tan seguro de la calidad de este producto que llegó a decir las siguientes plabaras: "estamos convencidos de la capacidad de SoftFactory/2000 de ofrecer un beneficio sustancial a los clientes. Por ello, Micro Focus está pidiendo a las campoañías, tanto en el Reino Unido como en España, que nos pongan a prueba, que nos den 50.000 líneas de código Cobol para que podamos proporcionarles nuestras soluciones sin coste alguno".

Figura 3.



MASTERS DE INFORMÁTICA

ENTORNO UNIX

LA OBRA DE REFERENCIA MÁS COMPLETA Y ACTUAL

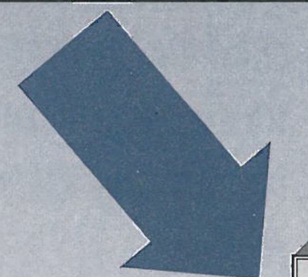
EL ENTORNO UNIX es:

- Uno de los sistemas operativos más extendidos.
- Es el candidato ideal para los servidores.

En esta obra podrá conocer en profundidad todas las particularidades de este sistema, contrastando la mayoría de sus variantes.

Tendrá asegurado un aprendizaje rápido y efectivo gracias al CD-ROM que acompaña el libro, el cual incluye:

- Todos los ejemplos que aparezcan en el texto,
- Programas de autor,
- Utilidades para Unix,
- Una versión completa de Linux.



Alejandro Novo



LA OBRA DE REFERENCIA MÁS COMPLETA Y ACTUAL

- La estructura de Unix
- El sistema multitarea
- Los Shell de Unix
- Los procesos y demonios
- Programación del Shell
- Unix e Internet

INCLUYE CD-ROM



PRECIO: 4.995 Ptas.

LIBRO + CD-ROM



c/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (Madrid)
Tel.: (91) 661 42 11* - Fax: (91) 661 43 86

ANÁLISIS Y DISEÑO OO

La técnica para modelado de objetos (OMT) (I)

Carlos Urbano

Dicen, que existió una nave espacial que surcó el espacio enviando datos a la tierra; y dicen también que hubo después una segunda nave que se estrelló antes de salir de la atmósfera. La primera nave utilizaba un software, que no se modificó en la segunda, que tenía los motores más potentes. Si el software hubiera sido OO, tal vez habríamos llegado a Marte antes.

En este mundo del software todo se hace corriendo, parece como si una de las principales acciones que los programas realizan hubiera sido del todo premonitoria en cuanto a las prisas con que todos los pasos del llamado proceso de desarrollo de software se realizan: *run* (ejecutar) que en otros contextos se traduce por correr. Y es verdad, todo se hace corriendo, los programas deben estar hechos para antes de ayer y las pruebas son escasas y poco exhaustivas. Los análisis suelen ser inexistentes y cuando llegamos a las empresas de desarrollo (o de servicios, que ahora está muy de moda eso de ofrecer a los informáticos como un servicio) nos encontramos con una página con dibujos garrapateados a bolígrafo donde aparecen algunos nombres y muchas rayas; eso suele ser el análisis.

Existen empresas donde utilizan a los analistas y les hacen trabajar como tales, donde los programadores reciben documentación sobre el proyecto antes de comenzar a desarrollar y no al contrario: la documentación la realizan ellos una vez que el proyecto ha sido entregado y aprobado por el cliente, en lo que en algunos sitios se denomina análisis detallado, pero la verdad es que es difícil encontrar alguna y cuando la encuentras suele tener algún nombre terminado en I+D que indica que no se corresponde con lo que sucede en el mundo real o es tan difícil entrar a trabajar en ella que desistes del intento hasta tener otros veinte años de experiencia.

El paradigma de la orientación a objetos se fundamenta en las entidades que conforman un sistema, no en las acciones que éstas realizan

Inmersos en este panorama nos hemos llegado a preguntar de qué sirve tanta metodología cuando nos introducen en la universidad - la mayor parte de las veces sigue siendo allí donde estudiamos materias que parecen no servir para nada más que para tener que aprobarlas, y cuesta mucho darse cuenta de nuestro error - en las asignaturas que tratan las distintas fases del desarrollo software. La mayoría de las veces estas metodologías están íntimamente relacionadas con la teoría de la orientación a objetos, aunque también existen muchas que se basan en la programación más tradicional. Las metodologías a veces conllevan un proceso de desarrollo y otras veces únicamente proponen una manera de abstraer el problema para solucionarlo de forma más fácil. Warnier, Bertini, Yourdon, Jackson y muchas otras son metodologías y notaciones para programación estructurada que han dejado paso a otras que incluyen dise-

ño de bases de datos relacionales (aunque alguna de ellas también contemple el diseño de bases de datos, no tiene un enfoque demasiado actual) y posteriormente, a las reinas de las metodologías que se utilizan el enfoque de la orientación a objetos. Y es aquí donde los métodos para conseguir modelos de los sistemas cobran su mayor relevancia.

El paradigma de la orientación a objetos

Si realizamos el análisis detallado de un proyecto informático de envergadura que no utilice OO, tendremos la oportunidad de dar a la orientación a objetos la importancia que se merece. En una maraña de funciones, módulos, variables globales, variables locales, librerías, accesos a ipcs (en Unix) como memoria compartida, semáforos o colas, bases de datos y archivos, normalmente nos perdemos sin remisión. Si intentamos entender el código que conforma las funciones de cada módulo, podemos tener mayor o menor fortuna según nuestra experiencia en la programación en el lenguaje determinado que utilicemos. Pero aunque entendamos qué hace una función, o muchas, probablemente perderemos el sentido de qué hace la aplicación.

Reuniendo un conjunto de operaciones que se realizan sobre unos datos tal y como se estructura la programación tradicional, no seremos capaces de desentrañar la funcionalidad del proyecto. Sólo cuando accedemos a las estructuras de datos, a la definición de las bases de datos y de sus archivos, a la definición de los recursos que se utilizan y a la forma en que se relacionan (por ejemplo al estudiar la memoria compartida entre procesos en Unix), podemos llegar a entender realmente qué objetivos persigue la aplicación. Las funciones por separado no nos dicen nada más que lo que cada una implementa y es en este detalle en lo que se basa el enfoque OO. Lo principal son las estructuras de datos y cómo se relacionan unos con otros, las operaciones que realizan o que se realizan sobre ellos ocupan un segundo plano y no definen la estructura del sistema.

Es de radical importancia entender esta diferencia fundamental existente entre las formas de concepción de un sistema ya que implica un cambio radical en la fase de análisis de la aplicación. El paradigma orientado a objetos se basa en esta visión del mundo, y por ende, de los sistemas que intentamos analizar y desarrollar. Son las entidades de este sistema, los objetos, los que forman el núcleo del sistema. Las operaciones les permiten modificar su estado y tener un comportamiento, pero no son el centro del enfoque. Además, esta idea conlleva un concepto muy importante en orientación a objetos. Los objetos se pueden reutilizar, ya que definen entidades, y, mediante los mecanismos propios de la OO (que suponemos conocidos por el lector), principalmente la herencia y el polimorfismo, podemos obtener almacenes de objetos que definan algunas operaciones generales y se especialicen a medida que sea necesario. Con estas ideas en mente, podemos llegar a entender por qué el desarrollo de software orientado a objetos requiere de un análisis mucho más exhaustivo de lo que el desarrollo tradicional necesitaba. Si un objeto (que estamos utilizando con la acepción de clase) está bien construido, es decir, sus características (miembros de datos en C++), sus operaciones (miembros funcionales en C++) y sus relaciones están bien definidas, esta clase se podrá utilizar en muchas aplicaciones distintas. La construcción del objeto implica la existencia de una fase de análisis por lo general de mayor duración que la que implica un enfoque no OO. Sin embargo, la robustez del diseño de la clase debe implicar también un ahorro considerable en el mantenimiento de la aplicación y en un nuevo desarrollo, ya que se pretende la reutilización posterior de las clases.

A pesar de esto, hasta hace relativamente poco tiempo, todos los esfuerzos realizados dentro de esta área han estado enfocados a los lenguajes de programación para realizar la implementación, más que a las fases de análisis y diseño. Esta tendencia está comenzando a cambiar y prueba de ello es la existencia en la industria de numerosas metodologías OO así como el desarrollo de nuevos estándares para modelado que se están realizando en la actualidad. El UML (*Unified Modeling Language*, Lenguaje Unificado para Modelado) está suponiendo un acicate para la industria, sus posibilidades como lenguaje para definir modelos son tan amplias que son muchas las empresas del sector que ya lo utilizan o lo están introduciendo en sus herramientas CASE. Con este artículo pretendemos continuar una serie que nos introducirá en las metodologías para desarrollo OO mediante el estudio de la OMT (*Object Modeling Technique,* Técnica para modelado de objetos) - sin extendernos demasiado, en un contexto eminentemente práctico como es *Sólo Programadores* - para conocer en próximos artículos el UML, que se convertirá en un estándar para modelado de sistemas informáticos.

El modelado y la abstracción del mundo

El desarrollo de software es un proceso que tiene distintas fases. La especificación de estas fases varía según los autores que definan el proceso. Sin embargo, casi todos los autores coinciden en cuatro fases de gran importancia: análisis, diseño, implementación y mantenimiento. En OO son las dos fases primeras, el análisis y diseño, las que tienen mayor trascendencia ya que es en estas fases donde se obtiene la esencia del sistema identificando los conceptos y su organización. La implementación en un lenguaje OO determinado es una parte importante pero el análisis determina el éxito del desarrollo. Es importante entender qué significa realizar un análisis: los analistas abstraen los conceptos de la aplicación para llegar a

determinar qué aspectos son relevantes, una vez que se han identificado los conceptos se pueden tratar los detalles de implementación.

Este proceso de abstracción del problema extrayendo los aspectos principales del sistema se realiza a través del llamado *modelado* de la aplicación: se construye un modelo de una parte del sistema basado en un enfoque determinado que nos interese, y obviando los puntos que no sean importantes desde este enfoque. Podemos construir tantos modelos como creamos necesarios, uno por cada enfoque que necesitemos para obtener una visión global del sistema lo más completa posible. La abstracción en el modelado es suficiente como para permitirnos centrar nuestra atención en distintos puntos por separado facilitando así la construcción de un todo que, si intentamos abarcar en toda su complejidad, nos resultaría imposible entender.

Los métodos para realizar modelado proponen diferentes maneras de acometer esta tarea de abstracción. Nosotros centraremos nuestra atención en una metodología para el desarrollo OO que incluye una notación gráfica para representar los modelos, la OMT, creada por James Rumbaugh, Michael Blaha y otros, que representa una de las metodologías más importantes de la actualidad. Las metodologías son muy útiles en el proceso de modelado ya que permiten construir modelos del sistema que implementar y añadir posteriormente detalles de implementación que completarán el sistema.

La Técnica para Modelado de Objetos (OMT)

Las metodologías conllevan un proceso de desarrollo de software diferente según la teoría que defiendan. La OMT diferencia cuatro etapas principales: el análisis, el diseño del sistema, el diseño de objetos y la implementación. Es posible aplicar los conceptos de la teoría de OO a todas las fases del desarrollo y debe ser posible traspasar las clases de una etapa a otra sin más que ir añadiendo detalles en cada una de las etapas. También es normal ir añadiendo clases completas en fases posteriores, sobre todo cuando éstas no aparecen en el sistema a modelar sino que son propias de la implementación, ya que es habitual necesitar estructuras de datos que estén relacionadas con los equipos informáticos, por ejemplo. Estas fases de desarrollo se materializan en la creación de tres modelos diferentes del sistema:

1. Modelo de objetos
2. Modelo dinámico
3. Modelo funcional

Los diagramas de objetos, de estados y de flujo de datos son representaciones gráficas de los distintos modelos que propone la OMT

El Modelo de Objetos [MDOO] *pretende obtener una representación de la estructura estática de los objetos que existen en el sistema*, así como de sus relaciones, mediante los llamados **diagramas de objetos**. Un diagrama de objetos es un grafo que tiene como nodos las clases y los arcos definen sus relaciones. El Modelo Dinámico abstrae los aspectos de un sistema que están sujetos a cambio. Especifican las formas de control del sistema mediante los llamados **diagramas de estados**, que son grafos donde sus nodos son estados y los arcos son transiciones entre estados que son provocadas por *sucesos*. El Modelo Funcional define las transformaciones que se producen en los valores de los datos mediante los **diagramas de flujo de datos**, que son diagramas cuyos nodos son procesos y cuyos arcos son flujos de datos. A lo largo de este artículo nos ocuparemos del Modelo de Objetos, que constituye la parte más importante de la técnica para modelado, y dejaremos para artículos posteriores el resto de los modelos. A medida que vayamos adentrándonos en los diferentes modelos iremos también adquiriendo nuevos conocimientos sobre los conceptos implicados en el análisis y el diseño.

El modelo de objetos

La razón por la que el modelo de objetos constituye la parte principal de la Técnica para modelado es obvia teniendo en cuenta que ésta se fundamenta en la teoría de OO. Una definición clara de las entidades que intervienen en el sistema es un paso inicial necesario para poder definir qué transformaciones ocurren en ellas y cuándo se producen estas transformaciones. Esta forma de pensar es inherente al paradigma OO donde las clases y su jerarquía determinan el sistema.

El modelo de objetos representa la parte estática del sistema, mediante los diagramas de clases y los diagramas de casos concretos

El lector debería tener un mínimo conocimiento sobre la teoría de la OO para que este artículo le resulte interesante, no obstante los conceptos más avanzados del modelado no son generales en OO, por lo que los explicaremos cuando sea de utilidad para el entendimiento del texto. Otros conceptos como objeto, clase, herencia, polimorfismo u ocultación deberían sernos familiares; si no es así,

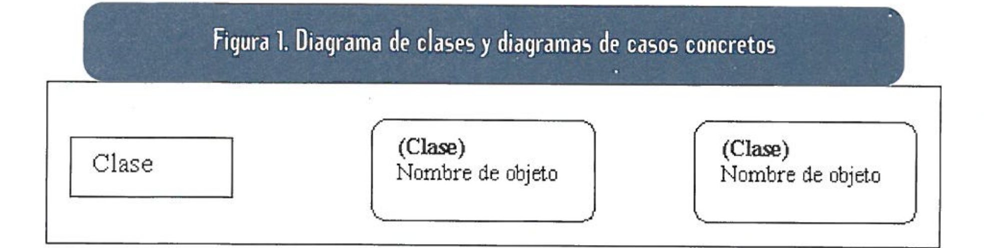


Figura 1. Diagrama de clases y diagramas de casos concretos

remitimos al lector a la bibliografía que se incluye en el artículo. El modelo de objetos está íntimamente relacionado con los conceptos mencionados, pretende mostrar los objetos, sus atributos o características y las relaciones que se establecen entre ellos de forma que se representan gráficamente y se facilita su posterior utilización en los otros modelos.

Los diagramas de objetos permiten representar gráficamente los objetos, las clases y sus relaciones mediante dos tipos de diagramas: los diagramas de clases y los diagramas de casos concretos (*instances*). Los diagramas de clases describen las clases que componen el sistema y que permitirán la creación de casos concretos, los diagramas de casos concretos describen la manera en que los objetos del sistema se relacionan y los casos concretos que existen en el sistema de cada clase. En los diagramas que componen este modelo se pueden representar los siguientes elementos del sistema: objetos y clases, atributos, operaciones, y relaciones o asociaciones.

Las clases y objetos en OMT

Los objetos y sus componentes se representan gráficamente en OMT de forma que es posible obtener una idea de los elementos que intervienen en el sistema estudiando el modelo. Los elementos y sus características con representación gráfica son los siguientes:

- **Objetos y clases.** En la figura 1 se representa un diagrama de clases y un diagrama de casos concretos. Un diagrama de clases es un rectángulo con el nombre de la clase en su interior, mientras que un diagrama de casos concretos se representa mediante un rectángulo con las esquinas redondeadas donde la clase aparece en negrita entre paréntesis y el objeto opcionalmente aparece en letra normal debajo de la clase.

- **Atributos.** Los objetos pertenecientes a una clase presentan unas características que en OMT se denominan atributos, un atributo de la clase figura puede ser número lados, si es sólida, color, número de vértices, etc. La figura 2 muestra la representación de los atributos dentro de los diagramas de clases y de casos concretos. Es muy importante en este punto no confundir los atributos, que son características que todos los objetos de una clase comparten, con otros objetos que pueden formar parte del objeto que estamos tratando. Existe un tipo de relación entre objetos denominada **agregación**, que permite indicar esta característica. Un objeto es agregado de otro cuando entre ellos existe una relación del tipo "forma parte" (*"is a part of"*). El objeto agregado contiene a otros objetos que a su vez forman parte de él y expresa la forma de los denominados objetos compuestos. La representación de este tipo de asociación se examina en el apartado sobre relaciones.

- **Clases abstractas.** En ocasiones puede ser de utilidad tener clases que definan propiedades y operaciones de forma general, dejando para sucesivos refinamientos la implementación concreta de las mismas. Una clase abstracta es precisamente una clase donde se introducen métodos y datos que se definirán en las subclases de la misma. Las clases abstractas siempre tienen que tener clases derivadas donde se especificarán las operaciones que no se hayan definido en la clase abstracta, nunca podrán tener objetos, se utilizan como clases bases o superclases de otras clases. La existencia de este tipo de clases facilita aún más la posibilidad de abstracción del modelo, dejando para posteriores refinamientos del problema la definición más exhaustiva de operaciones y datos. La implementación concreta de las clases abstractas depende del lenguaje OO, en C++, por ejemplo, las clases abstractas se consiguen con clases que definan métodos funcionales de tipo virtual, es decir, funciones virtuales. En OMT una clase abstracta se identifica cuando no tiene casos concretos (*instances*) pero una subclase suya sí los tiene. Para identificar un caso concreto en OMT se utiliza una línea punteada que une la clase con el caso concreto. Por ejemplo, una clase abstracta números podría definir una operación multiplicar que no se implementara hasta la definición de las subclases números enteros, números reales y números complejos, teniendo la información precisa en cada una de las subclases de la forma en que se multiplican los números.

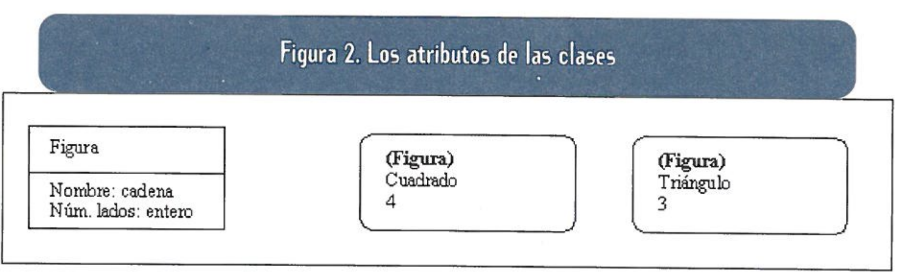
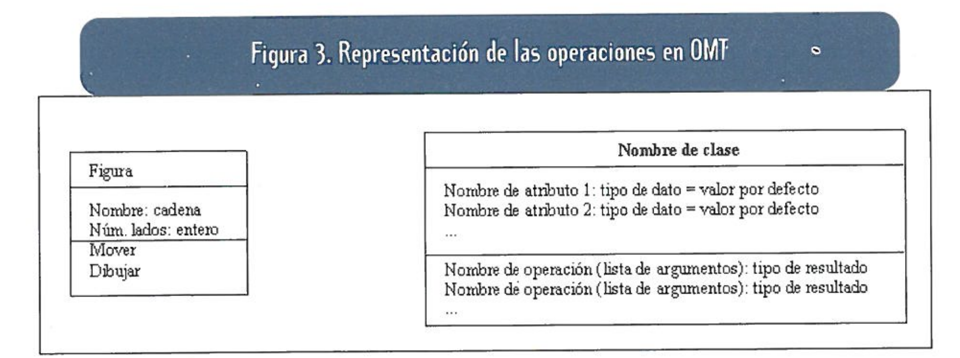


Figura 2. Los atributos de las clases

Figura 3. Representación de las operaciones en OMT



Las operaciones en OMT

Del mismo modo que los objetos en OMT se pueden representar las operaciones que se realizan sobre ellos o que éstos realizan sobre otros objetos del sistema. Los objetos realizan acciones sobre otros objetos y definen acciones que se realizan sobre ellos mismos. Los objetos de una misma clase comparten estas operaciones, aunque también pueden añadir otras nuevas que no se definan en su clase a medida que se especializa el objeto en otras subclases. También pueden redefinir las operaciones en estas especializaciones ignorando las definiciones realizadas en las superclases.

Los métodos son implementaciones concretas de las operaciones que se pueden realizar sobre objetos de distintas clases

Las operaciones pueden llevar implícito el objeto sobre el que se realizan o que realiza la acción, de forma que es posible tener una misma operación que se efectúe de manera distinta según el objeto sobre el que se aplique. Sobre la clase figura se puede realizar la operación *hallar el área*, para cada objeto de la clase esta operación se realiza de forma distinta, ya que las fórmulas para calcular el área de las figuras varían. La implementación de las operaciones para cada uno de los objetos diferentes (o subclases) se denomina en OMT *método*. Los métodos se implementan en cada una de las clases de forma específica para los objetos que representa.

Para conseguir este tipo de polimorfismo los lenguajes de programación proceden de distintas formas, por ejemplo en C++ además se permite la existencia de operaciones (funciones miembro) que se llaman de la misma forma pero se aplican a objetos diferentes simplemente variando su número de argumentos. En la figura 3 vemos la forma en que se representan en OMT las operaciones; éstas se indican en el tercio inferior del cuadro de la clase y opcionalmente cada nombre de la operación puede ir seguido por los argumentos (que van ente paréntesis separados por comas) y el tipo que devuelve el método, que se indica con dos puntos antes del tipo. Si no se indican los parámetros no se puede deducir que no los tenga, esta conclusión sólo se puede obtener explícitamente con una lista vacía (paréntesis vacíos). Tanto las operaciones como los atributos son opcionales en la representación.

Las asociaciones y sus características en OMT

Las relaciones entre clases determinan el comportamiento del sistema y constituyen una parte muy importante del mismo ya que mediante las relaciones definimos la forma en que los objetos se comunican, lo que también se conoce como comportamiento. Además las relaciones tienen una serie de características que son de interés para el modelado del sistema.

- **Relaciones.** En OMT se identifican a través de enlaces: *conexiones físicas o conceptuales entre casos concretos de objetos* [POAO]. Por ejemplo, en la clase personas el objeto profesor se relaciona con el objeto alumno de la misma clase mediante la relación "enseña a". La relación se establece con la acción "enseña a" entre los objetos relacionados. Los enlaces son a los objetos como las *asociaciones* a las clases. Una asociación en OMT abstrae un conjunto de enlaces con una estructura y un significado comunes. Si lo vemos gráficamente, una asociación en el dia-

Figura 4. Asociaciones entre clases y enlaces entre objetos

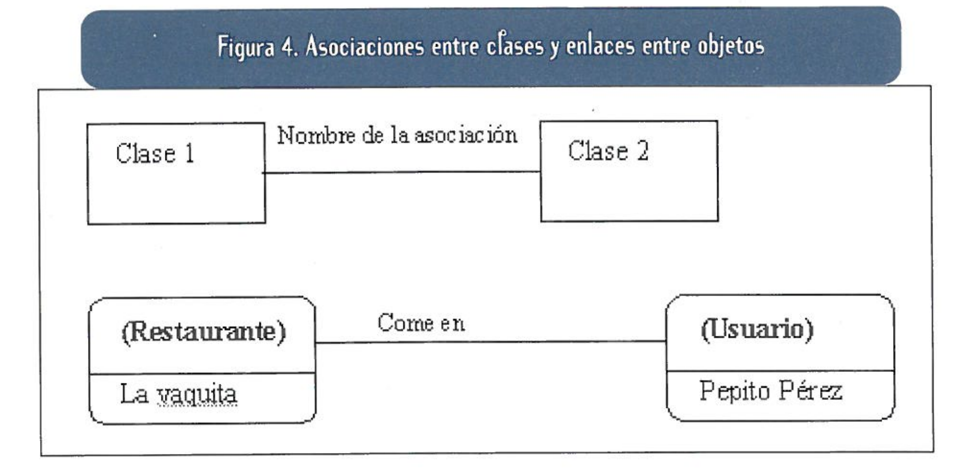
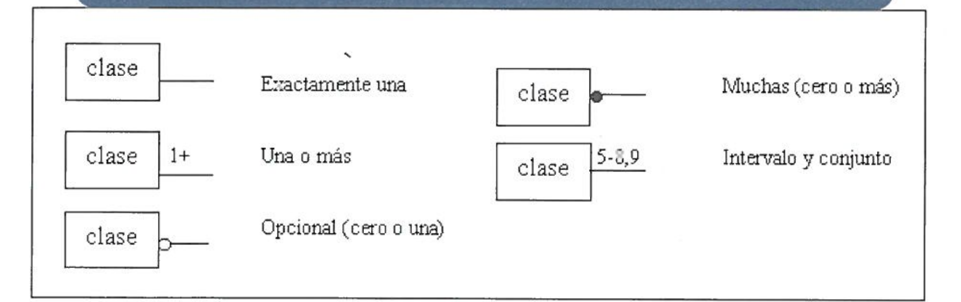


Figura 5. Representación de la multiplicidad en las asociaciones binarias.



grama de clases se corresponde con un conjunto de enlaces en el diagrama de casos concretos. Desde el punto de vista de implementación, una asociación es un puntero que apunta desde un objeto a otro. En OMT representamos las asociaciones como una línea que une las clases o los objetos relacionados y sobre la línea se indica el nombre de la asociación (que puede omitirse si la asociación es única y obvia). Véase figura 4.

- Relacionado con las asociaciones se encuentra el término **multiplicidad** que indica el número de casos concretos de una clase que puede relacionarse con otro caso concreto. Las relaciones más frecuentes son las binarias, aunque pueden darse de cualquier orden. La multiplicidad se indica en los diagramas de objetos con símbolos especiales al final de las líneas que indican las relaciones (asociaciones, como vimos antes). En las asociaciones binarias, una línea sin símbolos indica una asociación uno a uno, un círculo blanco indica cero o uno, y un símbolo negro indica muchos. También se pueden indicar números concretos de objetos relacionados con el número, el número con un + (suma) indica uno o más y se pueden indicar intervalos y conjuntos (5-8 indica de cinco a ocho y 5,8 indica cinco u ocho).

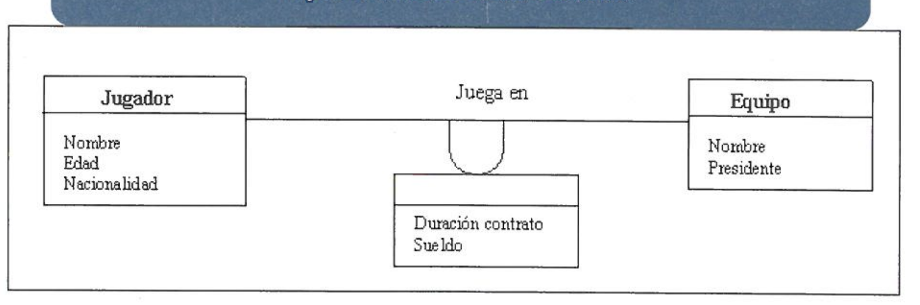
- En el caso de que las asociaciones no sean binarias, la notación gráfica para la multiplicidad varía ya que no se puede obtener una idea muy clara de la misma. En estas asociaciones se introduce el concepto de **claves candidatas** que nos sirven para limitar la multiplicidad de una asociación. Una clave candidata no es más que un *conjunto mínimo de atributos que define de forma única un objeto o enlace*. Es decir, mediante una clave candidata tenemos definido el objeto o el enlace con una serie de atributos de forma que se distingue del resto de objetos o enlaces. Las claves candidatas son restricciones y, por tanto, se representan como tales en OMT. Para encontrar una clave candidata en una asociación donde intervienen más de dos clases, debemos definir qué combinaciones de clases o enlaces en la asociación de los elementos definen la tercera clase o enlace de forma única.

- Los enlaces así como los objetos pueden tener **atributos**, que son propiedades de los enlaces de una asociación. La forma de representarlos en OMT es mediante un cuadro que se vincula a la asociación con un lazo. Los atributos de enlace deben ser propiedades de los enlaces. Los atributos de los enlaces pueden confundirse con los atributos de las clases, sin embargo, no podrían ser atributos de clases porque no caracterizan a las clases sino que caracterizan las relaciones entre ellas, por ejemplo, la duración de un contrato no se asocia sólo a un jugador, sino a la relación que establece con un equipo determinado. Véase la figura 6.

Las claves candidatas son restricciones y, por tanto, se representan como tales en OMT

- A veces resulta conveniente modelar las asociaciones como clases en lugar de como relaciones, cuando los enlaces pueden participar en asociaciones con otros objetos o están sometidos a operaciones. En estos casos, las asociaciones se representan con el cuadrado con el nombre de la asociación sus atributos y sus propias operaciones. También podemos encontrar en una asociación de objetos que pertenecen a una clase con multiplicidad "muchos" que deban estar ordenados. Esta característica se indica escribiendo "{ordenado}" en el lado del símbolo de muchos que se corresponde con la clase cuyos objetos deben estar ordena-

Figura 6. Atributos de una asociación.



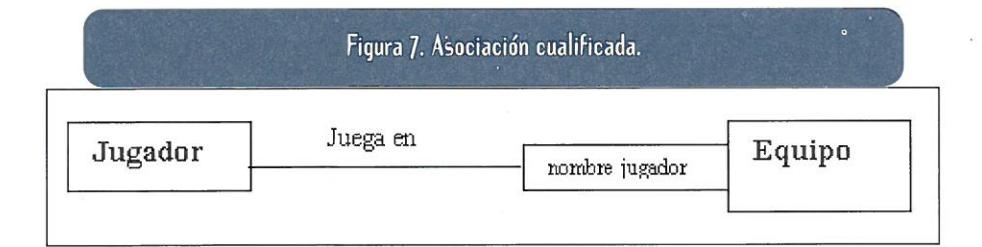


Figura 7. Asociación cualificada.

dos. Esta característica de los objetos es una restricción, ya que implica una condición que deben cumplir los elementos de la clase.

- Las asociaciones muchos a muchos y uno a muchos pueden ser **calificadas** mediante un elemento calificador que reduce el conjunto de objetos relacionados, indicando un subconjunto de la clase que se califica en la asociación. Este tipo de asociaciones se representan en OMT con un cuadro más pequeño que se interpone entre la clase relacionada y la línea que indica la asociación con otra clase (véase figura 7). Las asociaciones que se pueden calificar son las de multiplicidad uno a muchos y muchos a muchos. Por ejemplo, un equipo tiene muchos jugadores, pero un jugador sólo pertenece a un equipo (excepto cuando juega en su selección, caso que obviaremos en este momento).

Las relaciones de agregación representan relaciones del tipo "pertenece a" o "es parte de"

- Las asociaciones conectan clases u objetos que pertenecen a dichas clases, pero en ocasiones necesitamos restringir el conjunto de los objetos que se relacionan dentro de la clase. Mediante estas restricciones podemos especificar qué objetos se relacionan. Podemos conseguir este objetivo mediante los llamados **nombres de rol**. Un nombre de rol es un *atributo derivado de una clase cuyo valor es un conjunto de objetos relacionados.* Los nombres de rol se utilizan cuando las asociaciones se producen entre objetos de la misma clase ya que suelen producirse entre subconjuntos de esas clases. Una asociación binaria puede tener dos roles, uno por cada extremo de la asociación que son identificados por sus nombres y se representan como se aprecia en la figura 8. En un equipo juegan muchos jugadores, si el equipo es la selección nacional, sólo juegan los nacionales. Además, un jugador puede jugar en distintos equipos si es cedido.

- Las **relaciones de agregación** son para la OMT formas de asociación del tipo "es parte de", como tales se definen entre una clase agregado y una clase componente y se indican con un rombo en la parte de la clase que actúa como continente. Un ejemplo de agregación sería la que se establece entre las clases libro, capítulo y páginas. Un libro se divide en varios capítulos y cada capítulo tiene varias páginas. Esta relación se representa como se observa en la figura 9. Las relaciones de agregación se establecen en los llamados objetos compuestos que contienen otros objetos y éstos pueden ser de dos tipos: aquellos que tienen existencia física más allá del objeto agregado, como por ejemplo entre los objetos coche y sus componentes, ruedas, motor, parachoques, etc. y los que no pueden existir sin el objeto agregado, como por ejemplo los capítulos, que no existen sin el libro.

La herencia en OMT

En el paradigma de la orientación a objetos uno de los elementos más importantes es la herencia. La cualidad que permite que los objetos hereden las características (atributos) y las operaciones (métodos) dentro de una estructura jerárquica conlleva una serie de consecuencias de máxima relevancia a la hora de diseñar un sistema informático. Los objetos heredan un comportamiento que puede ser modificado, como ya conocemos, y unas estructuras de datos de forma que se permite y se facilita la reutilización de las clases y del código que implementa sus funcionalidades.

Ambos conceptos van unidos: herencia y estructura jerárquica, de forma que la herencia se produce por la existencia de una estructura entre los componentes del sistema y la estructura se consigue en la implementación del código a través de la herencia en los lenguajes OO.

La herencia está íntimamente relacionada con la forma concreta en que un lenguaje implementa la generalización, que es un término más abstracto.

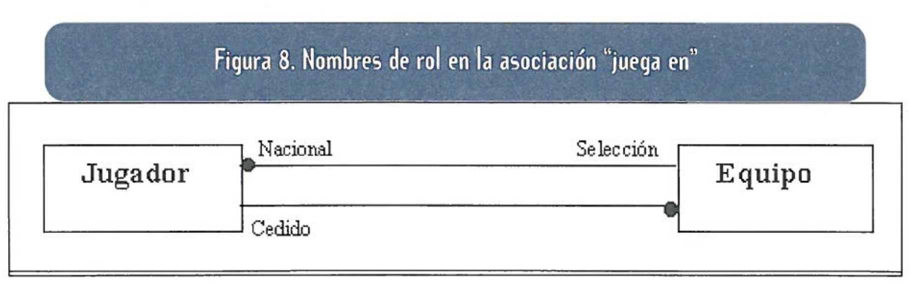


Figura 8. Nombres de rol en la asociación "juega en"

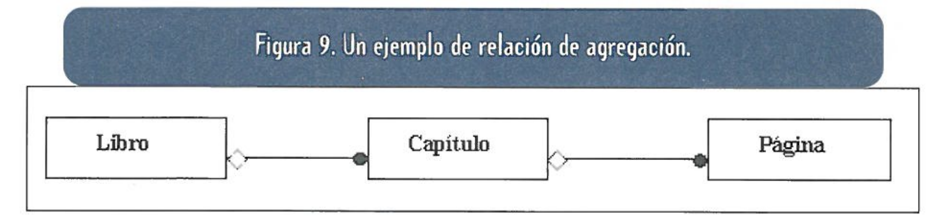


Figura 9. Un ejemplo de relación de agregación.

La generalización es la relación que existe entre una clase y las subclases que se derivan de la misma. A partir de una versión "en bruto" se generan versiones más especializadas de la misma que añaden características y operaciones. La superclase es la versión general y las subclases son especializaciones de la superclase. En la clase animal, las subclases pueden ser animal vertebrado, animal invertebrado, que a su vez pueden derivar otras subclases como animal mamífero, reptil, ave, etc. En OMT la relación de generalización se representa mediante un triángulo equilátero con un vértice orientado hacia la superclase. Los puntos suspensivos de la figura 10 indican que la clase animal puede estar compuesta por más clases, y también define una clase abstracta. Las generalizaciones pueden tener discriminadores que indican qué aspecto de la superclase está siendo utilizado para obtener subclases más concretas. Los discriminadores se indican con un nombre que aparece junto al triángulo, tal como podemos observar en la figura 11.

Herencia múltiple

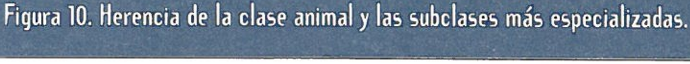
La herencia múltiple es una característica que algunos sistemas OO poseen, mediante la cual es posible que una clase herede de varias superclases al mismo tiempo, o de una clase que hereda de varias superclases. Sin embargo, la herencia múltiple aumenta radicalmente la complejidad de los sistemas que la implementan ya que la búsqueda de las operaciones (miembros funcionales en el lenguaje de programación C++) se dificulta cuando no se define en la clase derivada y hay que realizarla en las superclases.

La herencia permite crear componentes reutilizables que son de gran utilidad en el desarrollo de aplicaciones

El problema también se produce cuando existen distintas definiciones de la misma operación en distintas superclases, debido a que se producen conflictos que son resueltos de distintas formas según los lenguajes. En OMT la herencia se indica mediante el triángulo invertido y se distingue cuando existe herencia múltiple porque las subclases tienen una estructura donde existen varias superclases relacionadas con ellas. Es posible distinguir cuándo dos clases que heredan de una superclase se componen de objetos que no pueden pertenecer a la vez a las dos (generalización disjunta), mediante un triángulo sólido, de clases que pueden a la vez tener características de varias de las superclases de las que se derivan, que se representan con un triángulo hueco. La figura 11 ilustra este ejemplo, un animal se puede dividir en acuático y terrestre, y a su vez se pueden especializar en mamíferos, aves y peces, pero un mamífero puede ser a la vez acuático y terrestre con lo cual heredaría características de ambas clases. Este tipo de herencia es posible, pero no lo es cuando una clase se diseña de forma que hereda de dos clases que proceden de una generalización disjunta. La generalización según el medio en que vive el animal es disjunta.

Anulación de operaciones

Al implementar la herencia nos encontramos en numerosas ocasiones que las subclases redefinen operaciones que ya han sido definidas en las superclases. Las razones para esta nueva implementación de operaciones que existen en las superclases son variadas, a veces simplemente se le añaden nuevas acciones que van en consonancia con las nuevas características que añade la subclase; otras, se consigue optimizar las operaciones debido a que las subclases tienen características nuevas que lo permiten, y a veces se produce por una mala práctica de análisis donde no se prevén las operaciones de manera óptima.


Figura 10. Herencia de la clase animal y las subclases más especializadas.

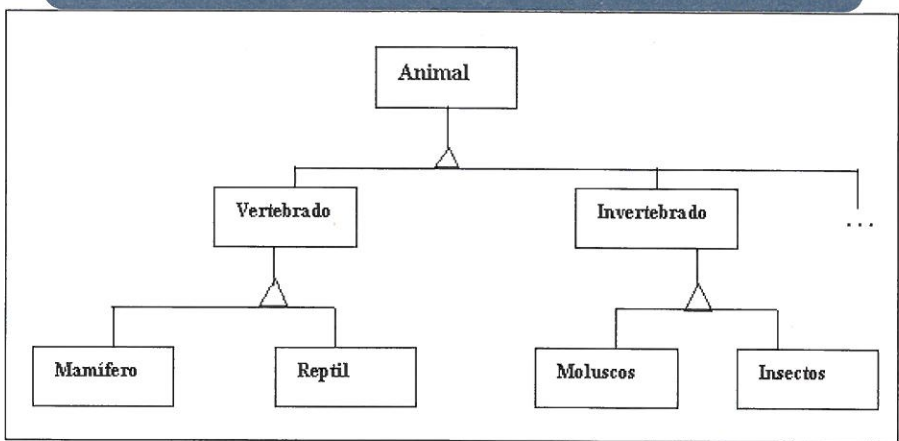
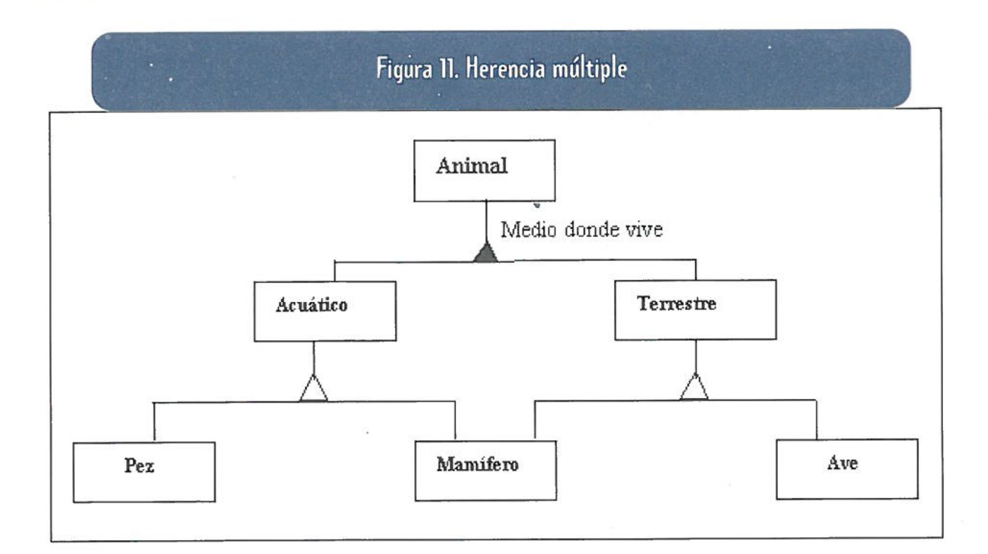


Figura 11. Herencia múltiple



Rumbaugh, en [MYDOO] propone algunas reglas útiles a la hora de implementar la herencia que minimizar los errores y maximizar la reutilización de código:

1. Las operaciones de consulta (es decir, que no modifican valores de atributos) se heredan por todas las subclases.
2. Las operaciones de actualización (que modifican los atributos) se heredan por todas las subclases y se añaden las nuevas operaciones para aquellas que añadan atributos.
3. Las operaciones de actualización que se realizan sobre atributos que tengan algún tipo de restricción o asociación, se bloquean para nuevas subclases.
4. Las operaciones no pueden volver a definirse para hacer que se comporten de distinta forma de cara al exterior, es decir, todos los métodos concretos de una operación deben tener el mismo protocolo.
5. Las operaciones se pueden refinan añadiendo comportamientos en las subclases.

Restricciones

El modelo de objetos contiene diferentes entidades como son los objetos, las clases, los atributos, los enlaces y las asociaciones. Cada una de estas entidades tiene una serie de características inherentes a su naturaleza dentro del sistema que estamos modelando. Las características definen su significado dentro del sistema y también sus limitaciones. Un objeto de la clase persona con atributos edad y peso, tiene unas limitaciones que derivan de su propio significado, por ejemplo, el atributo edad no podrá sobrepasar de 150 (no conozco a nadie que haya vivido más tiempo) y el atributo peso tampoco podrá sobrepasar los 200 kilos. Estas limitaciones se denominan en Análisis restricciones. Las restricciones pueden ser muy complejas y en este caso no pueden estudiarse en el modelo de objetos, sino que se especificarán en el modelo funcional. Las que intervienen en el modelo de objetos se representan mediante llaves junto a la entidad a que se refieran.

El modelado de objetos contiene las entidades: clases, atributos, enlaces y asociaciones

Cuando la restricción implica más de una clase, se indica mediante una flecha discontinua que une los elementos que se vean implicados en la restricción. Las asociaciones también pueden tener restricciones. La restricción se especifica mediante comentarios entre llaves. Un ejemplo de restricción son las clases ordenadas vistas anteriormente.

Agrupación de entidades

Los elementos que hemos estudiado en el Modelo de Objetos de la OMT se pueden agrupar para construir el modelo completo, así, las clases, las asociaciones y las generalizaciones forman lo que se denomina **módulo** y varios módulos forman el Modelo de objetos. En un módulo no se deben repetir los nombres de las clases y de las asociaciones, aunque se puede hacer referencia a la misma clase dentro de distintos módulos. También se definen las denominadas **hojas** que se utilizan para descomponer un Modelo de Objetos en unidades que podemos manejar, como podría ser una página impresa, aunque no necesariamente. Una hoja es una parte de un módulo que podemos manejar con facilidad, sea en el formato que sea.

Bibliografía

- *An introduction to Object-Oriented Analysis and Design*, Craig Larman, Prentice Hall, 1997
- Programación Orientada a Objetos, Luis Joyanes Aguilar, MacGraw-Hill, 1996
- Modelado y Diseño Orientado a Objetos, James Rumbaugh, Michael Blaha y otros, Prentice Hall, 1995
- *Object Oriented Design with Applicattions*, Grady Booch, Benjamin/Cummings, 1991
- C++, Manual de referencia, Herbert Schildt, MacGraw-Hill, 1995

Programación de DirectX con Delphi 2.0 (II)

DIRECTX

Constantino Sánchez Ballesteros

Datos de interés sobre DGC

DGC (Delphi Games Creator) debemos considerarlo un Sub-API de DirectX puesto que automatiza para nosotros las tareas más complejas de este último. Este hecho lo podemos ver mejor con un ejemplo:

Si tomamos como referencia la ayuda incluida en el DirectX SDK, para inicializar DirectDraw deberíamos realizar los siguientes pasos:

1. Crear un objeto DirectDraw en la variable de tipo IdirectDraw:

```
DirectDrawCreate2(nil, screen, nil);
```

2. Asignar a éste la resolución que deseemos para visualizar los gráficos:

```
screen.setdisplaymode(800,600,16);
```

3. El tercer paso consiste en seleccionar el comportamiento que tendrá nuestro programa en el entorno windows (pantalla completa, ventana...):

```
Screen.SetCooperativeLevel
   (application.Handle, DDSCL_FULLS-
   CREEN + DDSCL_ALLOWREBOOT +
   DDSCL_ALLOWMODEX +
   DDSCL_EXCLUSIVE);
```

4. Crear el *buffer* de vídeo principal:

```
video.dwSize := sizeof( video );
video.dwFlags := DDSD_CAPS
   +DDSD_BACKBUFFERCOUNT;
video.ddsCaps.dwCaps :=
   DDSCAPS_COMPLEX+DDSCAPS_FLI
   P+DDSCAPS_PRIMARYSURFACE;
video.dwBackBufferCount := 1;
screen.CreateSurface(video,
   lpddsprimary,nil);
```

5. Crear el *buffer* secundario:

```
BackCaps.dwCaps := DDSCAPS_BACK-
   BUFFER ;
lpDDSPrimary.GetAttachedSurface(back-
   caps,fsurface);
```

Por supuesto, estos pasos son los mínimos para inicializar el API, y si queremos características exclusivas de cada *buffer*, debemos indicarlo con otras tantas funciones. Con DGC, todo este código queda suprimido mediante la inclusión del componente DGCSCREEN en el formulario del proyecto. Os recuerdo que DGCSCREEN se encuentra en la paleta de componentes de Delphi dentro de la pestaña DGC, siempre que hayamos realizado la instalación que os detallé en el número anterior del curso. Esto os puede dar una idea de la comodidad y facilidad del uso de DirectX mediante los componentes DGC.

Si echáis un vistazo en el directorio BIN, creado por la instalación de DGC,

Continuamos con el estudio de las librerías DirectX de forma que empecemos a tomar contacto con las múltiples funciones y procedimientos que tenemos. En el artículo anterior vimos a grandes rasgos la estructura y el funcionamiento del API. Ha llegado el momento de ir comprobando la eficacia del mismo.

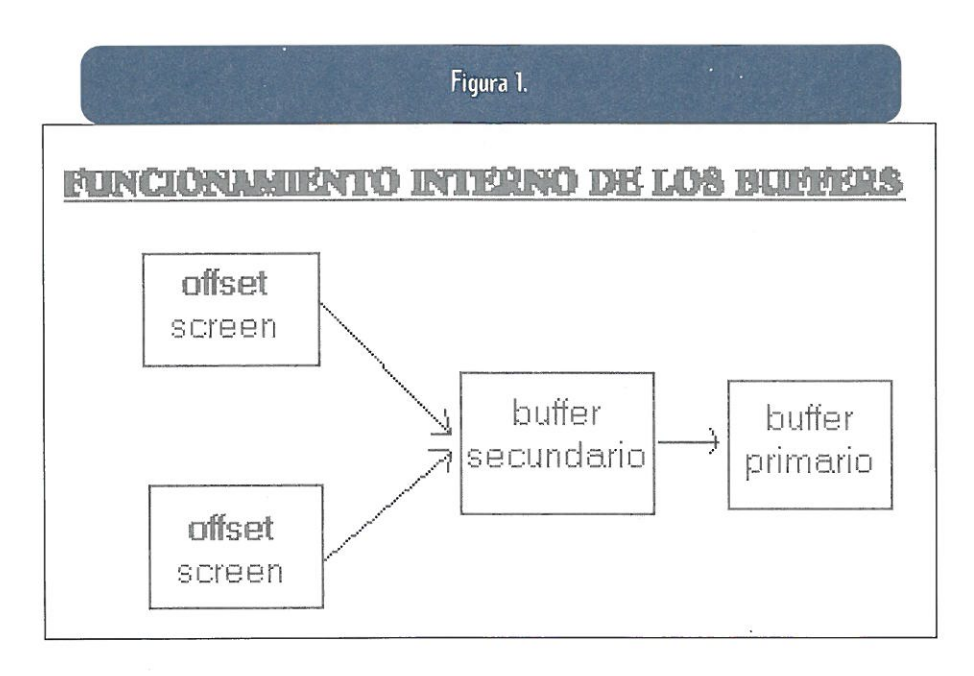
observaréis unas unidades que tienen nombres como DDRAW.PAS, D3D.PAS, D3DRM.PAS ... Todas ellas se encargan de definir las llamadas a las librerías DirectX, además de definir las correspondientes variables. Estas unidades han sido creadas por un grupo llamado **BlakeStone** y son fiel reflejo de los *headers* (unidades en Delphi) incluidos por Microsoft para la utilización de DirectX en lenguaje C o C++.

DGC se apoya en estas unidades para automatizar la mayoría de las tareas requeridas en la inicialización de diversos aspectos de DirectX. Puesto que DGC no soporta todas las características del API gráfico de Microsoft, podemos aventurarnos a crear nuestro propio Sub-API mediante la ayuda y ejemplos incluidos en el SDK. Aunque no es muy aconsejable, además de requerir unos amplios conocimientos de programación, podemos realizar pruebas con algunos procedimientos e instrucciones para crear tareas específicas. Un ejemplo puede ser la incorporación del modo *Clipper* de DirectX, que no es otra cosa que ejecutar los programas en ventana como si de una aplicación normal de Windows se tratara. Este aspecto todavía no se ha incorporado a DGC, aunque lo hará en breve.

Mediante DGC nos evitaremos mucho trabajo en las inicializaciones de los parámetros necesarios para cada tipo de objeto creado

El intentar crear una solución de este tipo nos obligará a estudiar los ejemplos incluidos en el SDK además de la ayuda (en Inglés) incorporada en el mismo. Esto conlleva tiempo y en muchos casos, la caída del sistema operativo al efectuar operaciones no permitidas.

Figura 1.



El ejemplo anteriormente citado lo he creado utilizando directamente las unidades creadas por **BlakeStone**, y no utilizo ninguna funcionalidad de DGC. Ya veis que para una simple inicialización del modo de vídeo, he requerido bastantes instrucciones en la construcción del programa. Es aquí donde se aprecia la potencia de DGC, puesto que con incluir el componente adecuado en el formulario, he realizado de un golpe todas esas tareas y sin programación de ningún tipo.

Depurado de programas

Cuando se trabaja a menudo con DirectX, podemos tener problemas al ejecutar nuestros programas puesto que podemos incurrir en operaciones ilegales al trabajar directamente con los *buffers* creados. Cuando compilamos un programa, si éste está bien declarado (sin errores en la sintaxis) el fichero EXE se creará sin problemas. Las complicaciones vienen cuando se ejecuta, y sobre todo en el modo *full screen* (pantalla completa). Cuando ejecutamos los programas en este modo (el que viene por defecto en DGC) se toman todos los recursos de Windows, y si se obtiene un error en tiempo de ejecución, un setenta por ciento de las veces nos será imposible restaurar el sistema del modo en que se encontraba anteriormente. Además, no podremos visualizar el tipo de error en que hemos incurrido.

Con esto quiero decir que debemos conocer bien las instrucciones de DGC con las que vamos a trabajar para no incurrir en problemas de este tipo, que además de restarnos tiempo, pueden estropear algún componente importante del sistema operativo y nos tocará reinstalar éste de nuevo. En mi caso, por intentar realizar operaciones adicionales a las permitidas por las funciones de DGC, me cargué la librería DIBENG.DLL que gestiona el módulo de dibujo en Windows, además del fichero MSGSVR.EXE. Con estos detalles, sólo quiero advertiros de los peligros que puede conllevar el efectuar operaciones independientes a DGC y que no utilicen sus componentes.

Si optáis por incluir alguna funcionalidad no implementada por DGC utilizando las unidades de **BlakeStone**, poned atención en los valores que se deben asignar a las propiedades antes de ser utilizadas.

Utilización de los buffers primario y secundario

Para poder trabajar correctamente con DirectX y efectuar nuestras operaciones gráficas sin restricciones, necesitamos utilizar los dos *buffers* que crea DGC al inicializarse:

1. Primario (Front)
2. Secundario (Back).

El *buffer* primario va enlazado directamente con la página de salida de vídeo, es decir, que los gráficos que vayan a éste serán visualizados instantáneamente en la pantalla del PC. Si sólo utilizáramos el primario para las operaciones gráficas, cuando quisiéramos efectuar animaciones en pantalla aparecerían efectos extraños como el famoso *flicker* (parpadeo) y muchos de los *sprites* efectuarían el movimiento a trozos y sin suavidad de ningún tipo.

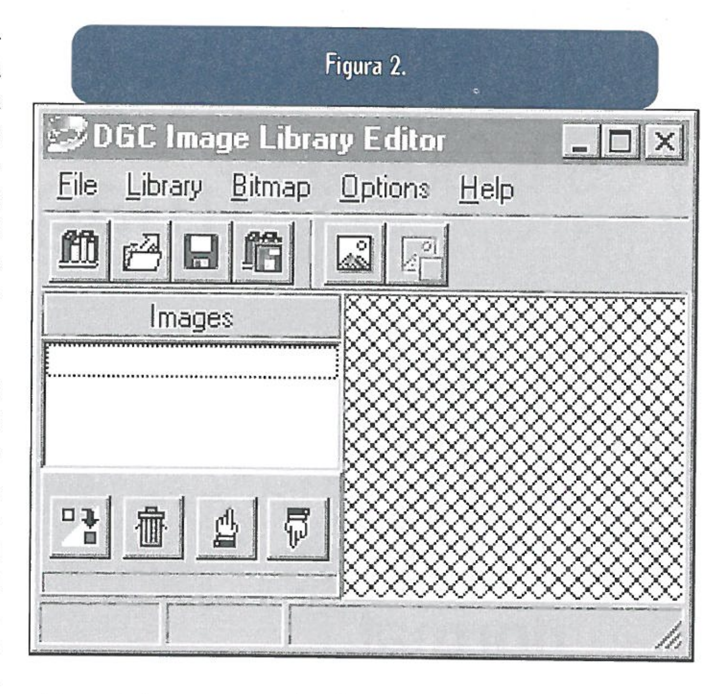
Para solventar este problema, se utiliza la técnica del doble *buffer*, que consiste en crear otro *buffer* al que enviaremos las operaciones gráficas, y una vez completado el proceso, reenviar éste al *buffer* principal. Con este sistema evitamos la carencia de refresco de la página principal de vídeo y las operaciones se efectúan con suavidad y sin *flicker* de ningún tipo.

También podemos crear más *buffers* secundarios, pero lo único que harán será ralentizar la representación gráfica, puesto que los gráficos irán pasando de un secundario a otro hasta llegar al primario.

IML (librería de imágenes)

Muchos os preguntaréis dónde guardar la información gráfica para luego pasársela al *buffer* secundario. DirectX exige que los gráficos sean guardados en diferentes *buffers* que funcionan de forma similar al primario o secundario. La diferencia entre ellos radica en que los *buffers* que guardan información gráfica para luego pasarla al secundario deben ser del tipo *offset-screen*.

Figura 2.



Podemos definir cuantos *buffers offset-screen* necesitemos, pero deben ser inicializados de una forma concreta para funcionar. Cuando decidimos pasar un gráfico al secundario, copiamos el trozo de imagen necesitado que se encuentre en un *offset-screen* y se lo enviamos. Después, sólo queda pasar el secundario al primario y visualizaremos la copia gráfica efectuada.

Con la IML creamos los gráficos necesarios que se incorporarán en nuestro programa, pero no se habrá creado ningún *offset-screen*. Esta tarea se encarga de realizarla el componente DGCIMAGELIB. Con él, crearemos de forma automática tantos *offset-screen* como gráficos haya en la librería de imágenes. Este componente inicializará los parámetros necesarios para su funcionamiento, al igual que lo hizo DGCSCREEN, y estaremos en disposición de efectuar las transferencias gráficas de un *buffer* a otro. A continuación veremos la utilización de la IML para poder incorporar nuestros gráficos al programa.

El ejecutable de esta aplicación corresponde al nombre de **ImgLib.exe** y debéis tener un acceso directo al mismo en la carpeta *programas* del botón *inicio* de Windows. Una vez abierto el programa, nos toparemos con una ventana tal que así:

Para crear nuestra primera librería gráfica iremos a la opción *file* del menú y seleccionaremos *new library*. También vemos que hay unos botones rápidos para crear nuevas librerías, abrir una existente, guardar librerías, editar paletas gráficas ... Seguidamente procederemos a cargar nuestro primer gráfico utilizando la opción *open* correspondiente al menú *bitmap*. Una vez seleccionado vuestro gráfico, procederemos a insertarlo en la librería. Para ello, iremos al menú *library* y pincharemos en *add image*.

Realizados estos pasos, tenemos creada una librería de imágenes preparada para su utilización con DGC. Tan sólo nos queda guardar el trabajo realizado en nuestro disco duro. Para ello, usaremos la opción de menú file/save library y daremos un nombre a la misma. El siguiente paso será incorporar el componente DGCIMAGELIB a nuestro formulario indicándole qué librería utilizar.

Lógicamente, para cualquier juego se necesita más de un gráfico y tendremos que cargar más imágenes a la librería a medida que necesitemos diferentes *frames* en cualquier animación o fondo gráfico. El proceso a seguir es el mismo que hemos efectuado anteriormente con la primera imagen. Con la opción de menú *bitmap/open* iremos insertando los siguientes gráficos que se incluirán en la librería.

números atrasados

SÓLO PROGRAMADORES

completa ya tu colección

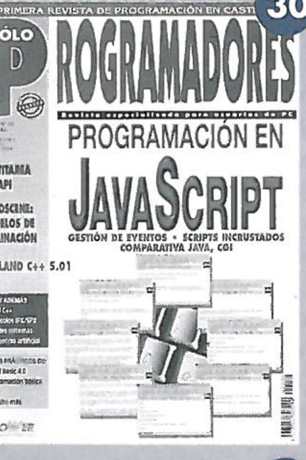




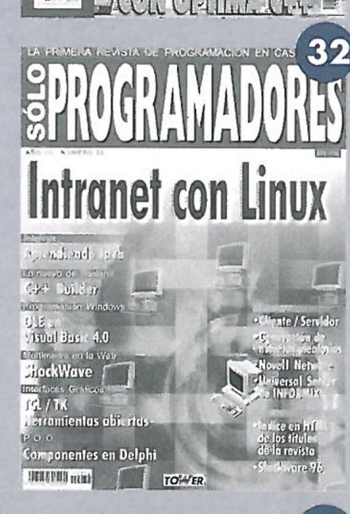




29
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CASTE
SÓLO PROGRAMADORES
BORLAND INTRABUILDER PROFESIONAL 1.0
COMUNICACIONES EN WINDOWS
PROGRAMACIÓN DE INTERFACES GRÁFICAS CON WINDOWS X (TCL/TK)
TOWER









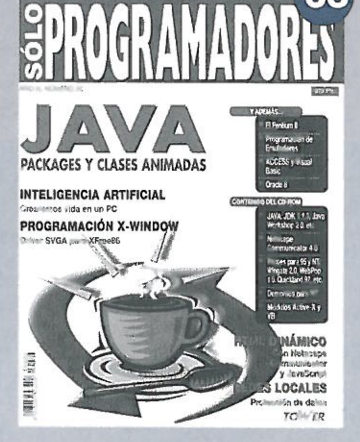
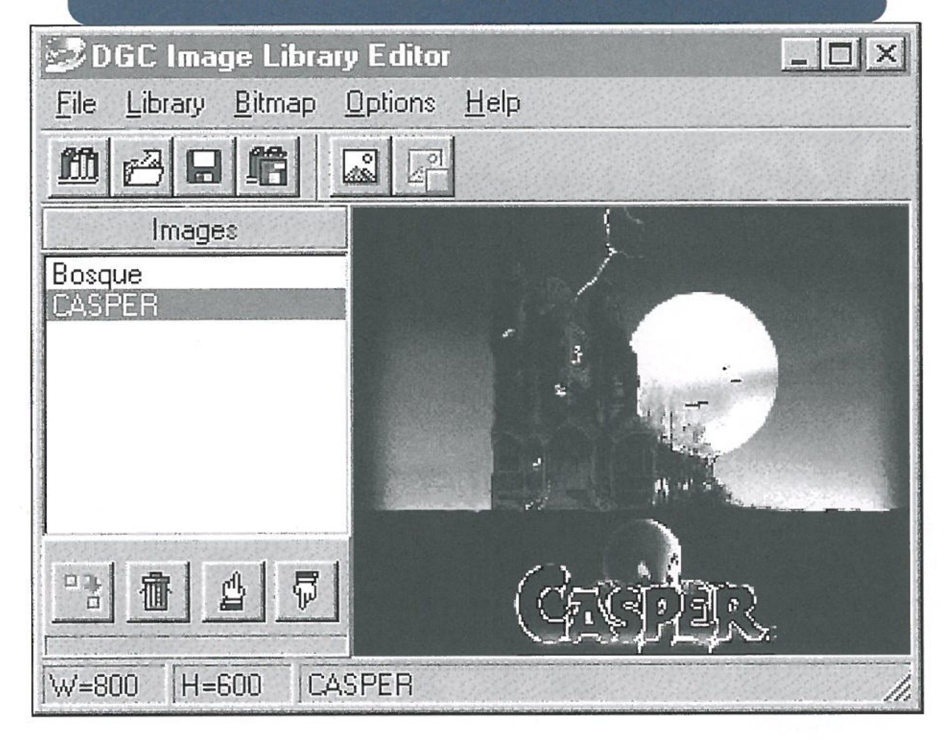


Figura 3. Librería de imágenes con 2 gráficos insertados.



Más adelante aprenderemos su utilización en DGC además de otras posibilidades que ofrece el programa.

Atención: En la librería sólo podremos cargar imágenes de hasta 256 colores puesto que todavía no se soportan los modos Hi-color en DGC.

Primer contacto en la programación

Llegó la hora de crear nuestro primer programa. En el CD de la revista podréis encontrar el proyecto completo que vamos a abordar. La función del programa consistirá en asignar un fondo gráfico a la pantalla y escribir un texto determinado sobre él. Una idea genial que han tenido los creadores de DGC ha sido el implementar la clase *tcanvas* para dibujar en los *buffers* creados con alguna nueva opción.

Lo primero que debemos hacer al abordar un proyecto con DGC es crear una nueva aplicación en Delphi e incrustar en el formulario el componente DGCSCREEN. De este modo tendremos a nuestra disposición todas las capacidades gráficas que ofrece DirectDraw. El siguiente paso será ajustar las propiedades del objeto a nuestras necesidades mediante el inspector de objetos:

- *BeepOnException*: Se utiliza para emitir un sonido cuando se encuentra algún error al ejecutar el programa. Estableceremos su valor a *false* para desactivarlo.

- *Debug*: Cuando ha ocurrido alguna excepción, con esta propiedad establecida en *true* se nos mostrará un mensaje indicativo mediante el programa *tracewin*.exe incluido en DGC. Nosotros lo dejaremos en *false*.

- *Displaymode*: Esta propiedad define la resolución de la pantalla para nuestra sesión en DirectDraw. Actualmente sólo son operativos los modos de pantalla completa y no se puede utilizar el modo ventana. De cualquier forma, DirectSound y DirectPlay no requieren de la clase *Tdgcscreen* y pueden ser usados en una aplicación regular Delphi (del tipo ventana).

Los modos de vídeo soportados por el tipo *Tdisplaymode* son los siguientes:

640x480x8, 320x200x8, 800x600x8, 1024x768x8, 640x400x8 y 320x240x8. El número 8 establecido en todos los modos significa que soporta colores de 8 bits (256 colores simultáneos).

Como veis, los modos soportados son bastante amplios y satisfacen las necesidades de cualquier usuario. En nuestro caso, estableceremos el modo de 640x480x8.

- *Flippingenable*: Si asignamos *True* a esta propiedad, un procedimiento *Flip* será creado y el evento *OnFlip* será llamado automáticamente cuando la aplicación esté ejecutándose (TApplication.OnIdle). Esta propiedad puede ser activada o desactivada en cualquier momento durante una sesión de DirectDraw. Nosotros asignaremos el valor *True*.

Una vez realizados estos pasos, pasaremos directamente a la ventana que contiene el programa e inicializaremos los valores necesarios. En el siguiente extracto podéis observar la primera cabecera de la unidad de nuestro proyecto en la que definimos los eventos y propiedades necesarias para manejar el programa.

Esto no tiene ningún secreto puesto que se realiza de la forma estándar de Delphi.

En el listado 1 se aprecia el componente *DGCScreen1* que se ha insertado automáticamente mediante Delphi al incorporarlo al formulario de nuestro proyecto. Además, aparecen los tres eventos con los que vamos a contar para la utilización del componente:

DGCScreen1Initialize (evento *OnInitialize*): Este evento es llamado des-

pués de DirectDraw y cuando se ha inicializado la librería de imágenes asociada. Es el lugar ideal para asignar los parámetros que requieren DirectDraw o la librería de imágenes correspondiente.

Formkeypress (evento *OnKeypress*): Corresponde al evento de formulario de Delphi *Onkeypress*. Ocurre cuando el usuario presiona una tecla.

DGCScreen1Paint (evento *OnPaint*): Este evento es llamado cuando *DGCscreen1* ha sido borrado por otra aplicación y necesita ser redibujado otra vez. El evento se activa cuando se cambian aplicaciones con ALT+TAB.

También se crea un procedimiento privado *Drawframe*, que se encargará de ejecutar las operaciones gráficas deseadas. La siguiente parte del listado (véase listado 2) es la implementación de la unidad y detalla las instrucciones incorporadas a los procedimientos enumerados anteriormente:

Después de efectuar las operaciones gráficas con la clase *Tcanvas*, debemos utilizar la función *Release* para asegurarnos que Windows no se cuelgue. El evento *OnPaint* sólo debería ser llamado cuando se pierden los *buffers* de DirectDraw. Este hecho ocurre si se cambia de aplicación con ALT+TAB. En nuestro ejemplo, si se da este caso, realizamos el dibujo en el *backbuffer* y lo volcamos a pantalla con el método *Flip*.

El método *Flip* traslada el *buffer back* al *front* para que las operaciones gráficas enviadas al primero se puedan visualizar en la pantalla. Os recuerdo que teniendo *FlippingEnabled* en estado *True*, esta tarea es realizada automáticamente cuando el evento *OnFlip* es llamado.

Cuando utilizamos la clase *Tcanvas*, las operaciones de dibujo suelen referirse de la siguiente forma:

```
Formulario_o_ventana_a_utilizar.Canvas.Operación
    _a_realizar(...)
```

En nuestro ejemplo, para poder dibujar en el *backbuffer* en vez del formulario, asignamos el componente DGCscreen1 agregado al formulario más el método *back* que indica la utilización del *backbuffer*:

```
DGCScreen1.Back.Canvas
```

Utilizando cualquier método de Canvas podemos realizar las operaciones gráficas sin restricciones y con acceso directo a memoria. En nuestro ejemplo hemos utilizado el método *textout* para sacar caracteres directamente a la pantalla.

Observando el listado, aparecen dos instrucciones determinadas:

1. **Fadepalettein()**: Este método restaura suavemente la paleta gráfica desde el tono negro hasta la paleta original contenida en los gráficos. El parámetro que se pasa entre paréntesis controla la velocidad con que se efectúa la operación (técnicamente se llama *fade*).

2. **Fadepaletteout()**: Realiza el efecto contrario al anterior, es decir, convierte la paleta original al color negro en todos sus registros.

Creación de un offset-screen sin necesidad de utilizar la IML

Anteriormente hemos podido comprobar a grandes rasgos la versatilidad de la IML y el componente DGCImagelib para crear *buffers offset-screen* que incorporen gráficos. Además de estas utilidades, podemos crear nuestros propios *buffers* sin necesidad de utilizar la librería y el componente. Para ello, DGC incorpora un método específico que crea un *buffer* de estas características:

```
CreateSurface (var Surface: TDGCSurface; W, H:
    Integer)
```

Listado 1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes,
      Graphics,
  Controls, Forms, Dialogs,DGC;

Type

  TForm1 = class(TForm)
  DGCScreen1: TDGCScreen;
                                      procedure
    DGCScreen1Initialize(Sender:
    TObject);
    procedure FormKeyPress(Sender:
    TObject; var Key: Char);
  procedure DGCScreen1Paint(Sender:
    TObject);

  private

  procedure DrawFrame;

 end;

var

  Form1: TForm1;
```

Este método creará un *offset-screen* del tipo TDGCSurface usando las variables W y H para definir el ancho y el alto de dicho *buffer*. Este *buffer* será asignado a la variable *surface*. Cuando se termina de definir la información gráfica, debemos destruir la variable *surface* utilizando el método *Release*.

Veamos un ejemplo práctico de la utilización de este importante método (véase listado 3).

En el ejemplo del listado 3 se definen 2 variables, *mysurface* y *bitmap1*. La primera se encargará de albergar la información gráfica al crear con ella un *buffer*. *Bitmap1* es la variable utilizada por la clase *Tbitmap* para cargar el fichero gráfico que se encuentre en nuestro disco

Listado 2

```
implementation

{$R *.DFM}

procedure TForm1.DGCScreen1Initialize(Sender: TObject);

begin
   //dibuja las operaciones gráficas en el backbuffer
   DrawFrame;

   //asigna la paleta gráfica al color negro para borrar el buffer
   DGCScreen1.FadePaletteOut(1);

//efectúa el Flip entre el backbuffer y la pantalla de video que visualizamos //(frontbuffer)
   //Todavía no se ve porque la paleta está en color negro
   DGCScreen1.Flip;

   //Restauramos la paleta a sus colores originales
   DGCScreen1.FadePaletteIn(100);
end;

procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);

begin
   //Si se presiona ESC (escape), quitamos la demo
   if Key = #27 then
   begin
     Close;
   end;
end;

procedure TForm1.DrawFrame;
var
   n: Integer;

begin
   with DGCScreen1.Back.Canvas do
   begin
       //dibujamos en el backbuffer mediante la clase Tcanvas y sus funciones de dibujo
       Brush.Style := bsClear;
       Font.Size := 24;
       Font.Color := clBlue;
       TextOut(110, 60, 'Delphi Games Creator');
       Font.Color := clTeal;
       TextOut(108, 58, 'Delphi Games Creator');
       Font.Size := 12;
       Font.Color := clYellow;
       TextOut(110, 150, 'Esto es una simple Demo para demostrar la utilidad');
       TextOut(130, 170, 'de la clase TCanvas incorporada a DirectDraw.');
       for n := 0 to 100 do
       begin
           Brush.Color := RGB(Random(255), Random(255), Random(255));
           FillRect(Rect(n, 250 + n, 640 - n, 480 - n));
       end;
       Font.Color := clRed;
       TextOut(225, 355, 'Presiona Escape para Salir');
       Release;
   end;
end;

procedure TForm1.DGCScreen1Paint(Sender: TObject);

begin
   //Este evento siempre es llamado cuando se cambia de aplicación con ALT+TAB
   DrawFrame;
   DGCScreen1.Flip;
end;

end.
```

duro. El método *Loadfromfile* es estándar de esta clase.

Una vez que hayamos creado el *buffer* con el método *createsurface*, usaremos *canvas* para dibujar en él el gráfico cargado anteriormente. Yo he utilizado una resolución de 800x600 para el *buffer* puesto que es lo que ocupa mi gráfico. Podéis asignar la resolución que queráis mientras se pueda albergar el gráfico en su totalidad. No debéis confundir el método *Draw* de *Canvas* con el referido a DGC. El método *Draw* de DGC se utiliza para transferencia gráfica entre *buffers*:

```
Draw (X, Y: Integer; SourceSurface:
      TDGCSurface; Transparent: Boolean)
```

Este método dibujará el *buffer* contenido en la variable *Sourcesurface* en las coordenadas X, Y asignadas. Si la variable *Transparent* es asignada *True*, el *buffer* será transferido de forma transparente utilizando como referencia el color especificado en su paleta gráfica. Para ello, utilizaremos la siguiente función:

```
mysurface.TransparentColor:=40;
```

En este caso, el color que se tomará como transparente será el número 40 de la paleta gráfica. Para saber qué número de color deseáis aplicar como transparente, necesitáis cargar el gráfico en un diseñador como **Paintbrush** o **Paint** (ambos de Windows) y visualizar la paleta para elegir el color correcto.

El método *Draw* es el utilizado en la mayoría de las ocasiones para realizar operaciones de *sprites* que iremos viendo a lo largo del curso.

Nuevas opciones en la versión 5.0 de DirectX

Puesto que estamos trabajando con DirectX, es conveniente estar al día en

todo lo referente a él y su *interface*. En las líneas siguientes veremos qué mejoras han implementado los desarrolladores de Microsoft a la última revisión del API.

La versión 5 nos ofrece más servicios y utilidades para la innovación de lo que fue la revisión 3.0. Debemos saber que no existe DirectX 4 puesto que se saltó de la versión 3 a la 5 directamente. La referencia al programador contiene nuevas funciones adicionales y servicios, aunque todas las aplicaciones escritas para versiones anteriores del API se compilarán y ejecutarán perfectamente sin efectuar ningún tipo de cambios.

En este apartado podremos comprobar nuevas áreas que son significativamente diferentes a revisiones anteriores. Estas diferencias son enumeradas por el tipo de componente de DirectX:

- **DirectDraw**

DirectDraw ha sido ampliado con nuevas capacidades de *Video-Port* que permiten a los programas controlar el flujo de datos desde una tarjeta capturadora de vídeo a un *buffer* de DirectDraw. Adicionalmente, DirectDraw HEL por fin explota el rendimiento de los procesadores *Pentium* MMX de Intel. DirectDraw testea automáticamente la presencia de un procesador MMX la primera vez que creemos un *buffer* en memoria. Sobre ordenadores que no posean *Pentium* (como los 486), este test puede causar un benigno error del tipo "Illegal Instruction" que será enviado por el depurador del compilador (debugger). De todos modos, esta excepción no afectará la estabilidad o el rendimiento de nuestra aplicación.

DirectDraw ahora soporta *buffers off-set-screen* más anchos que el primario. Anteriormente, sólo se permitían *buffers* más altos pero con un ancho igual o inferior al buffer primario. Podemos crear *buffers* tan anchos como necesitemos, permitiendo que el hardware de vídeo lo soporte.

También se ha incluído soporte para el novedoso puerto avanzado de gráficos (AGP). En sistemas equipados con AGP, podemos crear *buffers* en memorias diferentes a la incluída en la tarjeta gráfica. La estructura de capas empleada en DirectDraw (DDSCAPS) permite soporte para diferenciar entre la memoria local de vídeo estándar y la memoria no local usada mediante AGP. La información retornada por DirectX sobre las posibilidades *hardware* instaladas contiene miembros que envían información sobre operaciones de *blit* (frecuentemente *sprites*) utilizando memoria de video no local.

Mediante el uso de AGP, podremos obtener rendimientos de hasta 4 veces más que en sistemas PCI estándar.

- **DirectSound**

DirectSound incluye un nuevo *interface* (llamado *IksPropertySet*) que lo activa para soportar servicios ampliados ofrecidos por tarjetas de sonido y sus *drivers* asociados. Una gran novedad es DirectSoundCapture, que permite captura de sonido a través del micrófono u otros dispositivos usando el modelo COM de Win32, y en el futuro será ampliado para trabajar directamente con los *drivers* incluidos por el fabricante del *hardware*.

Las nuevas posibilidades de las tarjetas de sonido, como puede ser el uso de 64 voces polifónicas, también son soportadas por el nuevo *interface* de DirectSound.

- **DirectPlay**

DirectPlay incluye un nuevo *interface*, *IDirectPlay3*, que es exactamente el mismo que *IDirectPlay2* pero con nuevos métodos. De forma similar, *IDirectPlayLobby2* es una versión extendida de *IDirectPlayLobby*. Entre sus nuevas funcionalidades se incluye la habilidad en las aplicaciones de suprimir los diálogos del proveedor de servicios, creando accesos directos a la conexión, sesiones asíncronas para mantener y actualizar el listado de sesiones disponible, implementación del método *SetSessionDesc*, mejor soporte para sesiones protegidas por *password* (claves secretas), soporte para conexiones seguras al servidor y la capacidad de crear múltiples objetos DirectPlay.

Con estas nuevas características, DirectPlay se convierte en un poderoso API de comunicaciones.

Listado 3

```
procedure
    TForm1.DGCScreen1Initialize(Se
    nder: TObject);
var
  MySurface: TDGCSurface;
  bitmap1:tbitmap;

begin

    DGCScreen1.CreateSurface(MyS
    urface, 800, 600);
  Bitmap1 := TBitmap.Create;
  Bitmap1.LoadFromFile('leon.bmp');
  with MySurface.Canvas do
  begin
  Draw(0,0, Bitmap1);
  Release;
  end;
DGCScreen1.Back.Draw(0, 0, mysurfa-
    ce, false);
DGCScreen1.Flip;

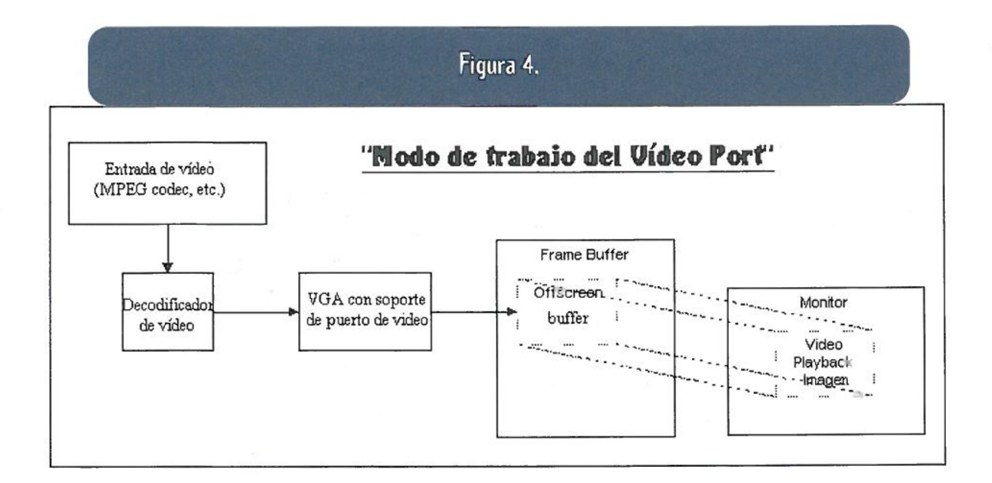
end;
```

- **Direct3D**

El modo inmediato de Direct3D (el más complicado de programar) soporta operaciones con primitivas sin tener que trabajar directamente con los famosos *Execute buffers*. El modo retenido de Direct3D amplía sus posibilidades mediante el soporte de interpolaciones que nos permiten crear transparencias en los colores, mover objetos suavemente entre posiciones, crear efectos de *morphing* (deformaciones) en los objetos 3D, y muchos más tipos de transformaciones. El modo retenido también soporta la carga de objetos 3D de forma progresiva; este sistema puede ayudarnos a elegir el nivel de detalle preferido al establecer progresivas bajadas (downloads) de objetos desde localizaciones remotas en la red.

La revisión del modo inmediato se ha creado para que sea más comprensible de lo que era antes, creándose tutoriales sobre el mismo además de detallar todos los entresijos del formato 'X' de objetos Diret3D.

Figura 4.



● **DirectInput**
DirectInput ahora permite interfaces tipo COM para la gestión de joysticks (entre ellos *game pads* y *flight yokes*) incluyendo el novedoso *Force Feedback* creado por Microsoft, además de los anteriores (ratón y teclado).

● **DirectSetup**
DirectSetup incluye un gran *interface* de usuario para crear fáciles instalaciones. Este *interface* se utiliza a través de una función que es pasada a DirectSetup antes de que se comience la instalación de los componentes DirectX y sus respectivos *drivers*. Esta función nos informa del estado actual de la instalación de nuestro programa. Esta información se puede utilizar para visualizar el estado de la instalación a través de un *interface* de usuario personalizado para nuestro programa. Además de todo esto, DirectSetup permite una vía para juegos multijugador que usen *DirectPlayLobby* para borrar su información creada en el registro de Windows.

● **AutoPlay**
No ha sido revisado en DirectX 5.0

Detalles de la nueva tecnología Video-Port

Como ya comenté anteriormente en las novedades de DirectX 5.0, se encuentra esta nueva tecnología del *video-port* que permite visualizar vídeo en tiempo real.

Un *Video-Port* es *hardware* implementado sobre un dispositivo de salida que permite acceso directo a un *buffer* incluido en otro de mayor tamaño llamado *frame buffer*, y todo a través del Bus PCI y la CPU. El acceso directo al *frame buffer* hace posible reproducir o grabar vídeo sin crear una carga de tiempo notable a la CPU. Una vez se tenga el fotograma en el *buffer*, la imagen puede ser visualizada en la pantalla como un *overlay* (superposición), una textura de Direct3D, o accedida por la CPU para tareas de captura u otros procesos. Los siguientes apartados dan una idea general sobre como trabaja interiormente la tecnología del *Video-Port*.

● **Data Flow (Flujo de datos)**
En un ordenador equipado con puerto de recepción de vídeo, los datos fluyen directamente desde la entrada de vídeo a través del decodificador. Una vez realizado el proceso, la tarjeta de vídeo recibe los datos decodificados y los envía al *frame buffer* para su manipulación directa. Normalmente, todos estos componentes existen juntos en la misma tarjeta gráfica, pero pueden aparecer como componentes *hardware* independientes que son físicamente conectados unos con otros. Un ejemplo de *Data flow* podría representarse gráficamente de la siguiente forma:

● **Video source (Dispositivo de vídeo)**
En el ámbito de la tecnología del *Video-Port*, un *video source* es estrictamente una entrada *hardware* de vídeo, como un MPEG *codec*, Zoom *Video-Port* u otro tipo de dispositivo. Estos dispositivos envían la señal al decodificador en una amplia variedad de formatos, incluyendo NTSC, PAL y SECAM a través de una conexión física.

● **Video Decoder (Decodificador)**
Un decodificador es un componente *hardware*. El trabajo de este dispositivo es descifrar la señal enviada por el *video source* y enviarla a su vez al *Video-Port* mediante un formato de conexión preestablecido. El decodificador posee una conexión física con el *Video-Port* implementando sus servicios a través de un *minidriver*. El decodificador es el responsable de los datos de vídeo enviados y sincroniza la información transmitida al *Video-Port*.

● **Video-Port**
Como en los anteriores componentes del envío de *data flow*, es otro dispositivo *hardware*. El *Video-Port* habita en las típicas tarjetas VGA y tiene acceso directo al *frame buffer*. El *Video-Port* recibe la información enviada por el decodificador, la procesa y la coloca en el *frame buffer* para ser visualizada. Durante este proceso, el *Video-Port* puede manipular la imagen recibida permitiendo efectos como *scaling* (ampliaciones de imagen), *shrinking* (disminuciones de imagen), control de color o servicios de *cropping* (recortes de imagen).

● **Frame Buffer**
El *frame buffer* acepta los datos de vídeo provistos por el *Video-Port*. Una vez recibidos, la aplicación puede manipular de forma programada los datos de la imagen efectuando operaciones de *blit* (transferencias de imágenes) de un *buffer* a otro, o mostrar la imagen como *overlay*. Esta última función es la más empleada de todas.

Con las posibilidades de la tecnología del *Video-Port*, no es descabellado pensar en la posibilidad de jugar vía red viendo las caras de los diferentes jugadores a los que nos enfrentamos, además de oír su voz, y lo que es mejor, todo ello en tiempo real.

WWW

Última versión del protocolo IP

Carlos Alvaro González
Sympha@hotmail.com

La versión 6 de este protocolo amplía el número de bytes para las direcciones, de 4 a 16, y ha dado en denominarse IPng (Internet Protocol Next Generation) o, simplemente, IPv6.

La familia de protocolos TCP/IP, desarrollados a principios de los años setenta en torno a la red militar ARPAnet, ha demostrado su eficiencia desde entonces hasta nuestros días, soportando el crecimiento exponencial que sus implementaciones han sufrido.

Actualmente, manejamos la versión 4 del Internet Protocol (IPv4), cuyas especificaciones se encuentran recogidas, como es habitual en todo lo relacionado con la Internet, en su correspondiente "Request For Comment", concretamente, el RFC 791, publicado en 1981.

La próxima versión de IP ha demostrado ser, con la experiencia vivida hasta el momento, una actualización de protocolo robusta y fiable

La versión 4 de IP utiliza cuatro octetos para definir la dirección de una máquina, estructurando estas direcciones en tres clases: A, B y C, (más otras dos, D y E, de propósito específico). De esta forma, la clase A utiliza los ocho primeros bits para definir la red, y los veinticuatro restantes para el número de host. La clase B, emplea dieciséis bits para la red y dieciséis para el host, y la C, veinticuatro para la red y ocho para el host. Así, en la práctica, se dispone de 254 posibles hosts para cada red del rango asignado para la clase C, que es la más utilizada habitualmente por las organizaciones que basan sus sistemas en TCP/IP.

La nueva versión

Las novedades aportadas por la próxima versión IPv6 frente a la anterior IPv4; (la versión IPv5 nunca fue más allá de una versión "de laboratorio"), abarcan varias características pensadas para dar soporte a las nuevas funcionalidades de las tendencias que, actualmente, se perfilan en el desarrollo de las comunicaciones TCP/IP.

La primera y más llamativa de estas novedades es la ampliación, de 32 a 128, en el número de bits de dirección. Según algunos estudiosos del tema, esto equivaldría a la posibilidad de poder direccionar la friolera de mas de

Los problemas derivados de la gran explosión del mundo Internet, en cuanto al direccionamiento IP de las máquinas, obtienen su solución mediante la entrada en vigor de la nueva versión del protocolo IP de red.

665.000.000.000.000.000.000.000 máquinas por cada metro cuadrado de la superficie terrestre. Se trata, por supuesto, de un cálculo "en bruto", sin establecer ninguna metodología de estructuración sobre el uso de las direcciones. Pero, aún haciéndolo, el número de ellas resultante por metro cuadrado, resultaría abrumador. (No menos de 1.564 posibles direcciones distintas, sometiéndolas a los planes más restrictivos).

El estudio previo

El organismo de la Internet responsable del cambio de versión, es el IETF (Internet Engineering Task Force), a través de un grupo de trabajo denominado IPng Working Group, encargado de obtener la solución final.

El Ipng Working Group reunió las características más relevantes de los estudios realizados, como el TUBA (TCP-UDP Bigger Addresses), el CATNIP (Common Architecture for The Next Internet Protocol), o el SIPP (Simple Internet Protocol Plus), para elaborar el IPng.

IPv6 multiplica por cuatro el número de bytes utilizados para el direccionamiento

Según se desprende del RFC 1726, donde se recogen los criterios técnicos para la elección de la nueva generación del Internet Protocol, el modelo resultante debería tener una serie de características precisas, a cuál más importante.

La primera y fundamental es la de que no supusiese ningún trauma el hecho del cambio de versión. Lograr la compatibilidad con la versión anterior, introduciendo cambios tan profundos en el sistema, sería una tarea difícil.

La próxima generación del protocolo debería satisfacer igualmente requerimientos de evolucionabilidad, escalabilidad y automatización, al tiempo que ser adecuado para los diferentes medios de transmisión y distintas topologías de red. También aparece una nueva característica a la que, hasta ahora, no se le había conferido demasiada importancia: la seguridad en la transmisión.

Funcionalidades

La evolución de Ipv4 a IPv6 conlleva, por una parte, la desaparición de funciones que no se utilizaban y, por otra, la aportación de nuevas funciones divididas en varias categorías:

1. Capacidades expandidas de routing y direccionamiento: aquí entran, tanto la ampliación de 32 bits a 128, como la adición de un campo de alcance ("scope field") para las direcciones multicast.

2. Direcciones "anycast": se trata de un nuevo tipo de direcciones para identificar grupos de nodos donde hay que distribuir un paquete enviado a una dirección de multicast. El uso de direcciones anycast permite a los nodos el poder controlar el camino por donde fluye el tráfico de paquetes.

3. Simplificación del formato de cabeceras: algunos de los campos existentes en IPv4 han sido eliminados o convertidos en campos opcionales. Esto se hace para reducir el coste de procesamiento; es decir, sólo se empleará tiempo de procesamiento en aquellos paquetes que incluyan dichos campos porque, efectivamente, los utilicen. A pesar de que el nuevo rango de direccionamiento es cuatro veces superior al actual de la versión 4, el tamaño de la cabecera IP completa de la versión 6 es sólo el doble.

4. Perfeccionamiento del soporte para opciones: las opciones en la cabecera IP permiten su codificación, con lo que se alcanza un mecanismo de forwarding más eficiente, que las limitaciones derivadas de la longitud de los campos de opciones sean menos restringentes, y que la flexibilidad para introducir nuevas opciones en el futuro sea mayor.

5. Capacidades de "calidad de servicio": es una nueva característica que permite habilitar el etiquetado de paquetes hacia un particular flujo del tráfico, para los cuáles el emisor requiere un manejo especial distinto del habitual por defecto.

6. Autentificación y privacidad: IPng incluye la definición de extensiones que proveen de soporte para la autentificación, la integridad de los datos y la confidencialidad. Estas características se incluyen como elemento básico en cualquier implementación de la nueva versión del protocolo.

La nueva cabecera IP

El encapsulamiento del paquete proveniente del nivel de transporte, se lleva a cabo por parte del nivel de red mediante la adición de 320 bits, (el doble que en la versión 4), estructurados en los ocho campos siguientes:

- VERSION (4 bits): este campo tiene el mismo significado que en la versión 4, y tendrá el valor "6" para IPv6.

- PRIORITY (4 bits): al igual que en la versión 4, este campo permitirá la existencia de algoritmos capaces de

Tabla 1.

ip	0	IP	(Internet Protocol, Pseudo Protocol Number).
icmp	1	ICMP	(Internet Control Message Protocol).
igmp	2	IGMP	(Internet Group Multicast Protocol).
ggp	3	GGP	(Gateway-Gateway Protocol).
tcp	6	TCP	(Transmission Control Protocol).
egp	8	EGP	(Exterior-Gateway Protocol).
pup	12	PUP	(PARC Universal Packet Protocol).
udp	17	UDP	(User Datagram Protocol).
hmp	20	HMP	(Host Monitoring Protocol).
idp	22	IDP	(Internet Datagram Protocol).
rdp	27	RDP	("Reliable Datagram" Protocol).
hello	63	HELLO	(HELLO Routing Protocol).
ospf	89	OSPF	(Open Shortest Path First Routing Protocol).

(Para implementaciones particulares de IP, consultar el fichero "protocols" del sistema).

determinar qué tipo de tráfico es de mayor importancia que otro. De esta manera, en caso de saturación de las comunicaciones, se puede mantener la preponderancia de un tipo de mensajería sobre otro.

- FLOW LABEL (24 bits): sirve para indicar tratamientos especiales en las transmisiones de datos. (Calidad de servicio).
- PAYLOAD LENGTH (16 bits): es un entero sin signo que contiene el número total de bytes del datagrama (sin contar la cabecera).
- NEXT HEADER (8 bits): identifica el tipo de cabecera inmediatamente siguiente a la cabecera IPng. Los valores utilizados son los mismos que en el campo "Protocol" de IPv4. Los más comunes de estos valores son los que muestra la tabla 1.
- HOP LIMIT (8 bits): es un entero sin signo, que se decrementa a medida que un nodo reenvía el paquete. El paquete sería descartado si este contador llegase a cero.
- SOURCE ADDRESS (128 bits): dirección IPng del emisor inicial del paquete.
- DESTINATION ADDRESS (128 bits): dirección IPng del próximo receptor del paquete (no necesariamente el receptor final).

IPng extendido

La versión 6 de Ip incluye un mecanismo mejorado de opciones sobre la versión 4. Ahora, las opciones se colocan en extensiones de la cabecera separadas, localizadas entre la cabecera IPng y la cabecera del nivel de transporte de cada paquete.

IPng satisface requerimientos de evolucionabilidad, escalabilidad y automatización, al tiempo que se preocupa de la seguridad en la transmisión

Con esto, la mayoría de las extensiones de cabecera no son examinadas ni procesadas por los routers intermedios, sino sólo cuando el paquete llega a su destino final. Con esto se consigue un mayor rendimiento del tráfico sobre la versión actualmente en uso, que obliga al examen de estas opciones por estos elementos intermedios de encaminamiento.

Otra ventaja de este sistema consiste en que no se obliga a una longitud fija para estas opciones que, principalmente, se usan para mecanismos de autentificación y encapsulación de seguridad.

Con objeto de mejorar el rendimiento en el manejo de opciones subsiguientes y en el transporte, las opciones IPng siempre constituyen un múltiplo entero de ocho bytes.

Las cabeceras de extensión actualmente definidas lo están para los siguientes conceptos:

- Routing. (Routing extendido).
- Fragmentation. (Fragmentación y reensamblaje).
- Authentication. (Integridad y autentificación; seguridad).
- Encapsulation. (Confidencialidad).
- Hop-by-Hop Option. (Opciones especiales que requieren procesamiento en cada salto del paquete).
- Destination Options. (Información opcional que debe ser examinada en el nodo de destino).

Direccionamiento IPng

Como hemos dicho, las direcciones IPng son de 128 bits de longitud, por tanto, IPng soporta 2 ^ 128 direcciones frente a las 2 ^ 32 de IPv4.

El cálculo es el siguiente: 2 ^ 128 = 340.282.366.920.938.463.463.374.607.431.768.211.456

Tabla 2.

Allocation	Prefix(binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP Allocation	0000 001	1/128
Reserved for IPX Allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider-Based Unicast Address	010	1/8
Unassigned	011	1/8
Reserved for Neutral-Interconnect-Based Unicast Addresses	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link Local Use Addresses	1111 1110 10	1/1024
Site Local Use Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

Estas direcciones se clasifican en tres tipos:

1. Direcciones Unicast, que identifican a un único interface de red.

2. Direcciones Anycast, que identifican a un conjunto de interfaces de red. De esta forma, un paquete enviado a una dirección anycast, será enviado a uno de los interfaces de dicho conjunto.

3. Direcciones Multicast, que identifican a un grupo de interfaces, de tal manera que, un paquete enviado a una dirección multicast, se distribuye a todos los interfaces del grupo.

Una de las variaciones más importantes de esta versión de IP, con respecto a la anterior, es que dejan de existir las direcciones broadcast. Esta funcionalidad es asumida ahora por el direccionamiento multicast.

El tipo específico de dirección IPng se indica por una serie de "bits de guía" (leading bits). La tabla 2 muestra el tipo de dirección en función del prefijo de la misma, y la fracción del espacio de direccionamiento utilizado para cada tipo. Aproximadamente, el 15% del espacio está inicialmente asignado, mientras que el 85% se reserva para su uso en el futuro.

Direcciones unicast

Dentro del espacio asignado para direccionamiento unicast, se establece una subdivisión de la siguiente manera:

1. Provider Based Unicast Addresses. Se usan para comunicación global siguiendo el siguiente esquema: Prefix: si los tres primeros bits son 010, se identifica a esa dirección como Provider Based Unicast Address. Registry ID: identifica la dirección Internet asignada a un organismo registrador de direcciones Internet para proveedores de direcciones. Provider ID: identifica al proveedor de direcciones, que obtiene su rango de direcciones del organismo registrador para, a su vez, otorgarlas a sus subscriptores. Subscriber ID: son los multiples subscriptores que han obtenido sus redes del proveedor de direcciones. Subnet ID: identifica a una red física específica. Interface ID: es el identificador del interface.

Las nuevas capacidades de "calidad de servicio" permiten al emisor requerir tratamientos especiales para sus paquetes

2. Local-Use Addresses. Son direcciones unicast que tienen sólo un ámbito local de routing, bien dentro de

una subnet o de una red de subscriptor. Hay dos tipos de Local-Use Addresses definidas: las Link-Local, que se usan para enlaces muy temporales y con un proposito específico, tal como autoconfigurarse con una dirección válida, y las Site-Local, que se usan para enlaces permanentes o semi-permanentes, en instalaciones que vayan a estar aisladas de la Internet.

Este último tipo de direcciones permiten, a las organizaciones que todavía no están conectadas a la Internet, operar sin necesidad de solicitar un prefijo de dirección válido del espacio global de direcciones. Posteriormente, cuando esta organización quiera conectarse a la Internet, podrá usar su identificador de subnet y su identificador de interface en combinación con los identificadores de registro, proveedor y subscriptor que adquiera.

El formato de estas direcciones es el siguiente:

a) Link-Local-Use Addresses

Prefix: (10 bits) 1111111010.
n bits: un número de bits a cero, que rellenan hasta llegar al Interface ID.
Interface ID.

a) Site-Local-Use Addresses

Prefix: (10 bits) 1111111011.
n bits: un número de bits a cero, que rellenan hasta llegar al Subnet ID.
Subnet ID.
Interface ID.

Direcciones embebidas en direcciones IPv4

El mecanismo de transición de la versión 4 de IP a la 6 esta especialmente preparado para que, tanto los routers como los hosts puedan recibir datagramas IPv6 según la estructura de IPv4.

Los nodos IPv6 emplean direcciones a modo de unicast, donde los primeros 96 bits están a cero y los 32 restantes forman la dirección IPv4.

Los nodos que no soportan IPv6 son mapeados a direcciones IPv6, donde los primeros 80 bits estan a cero, los 16 siguientes a 1, y los 32 restantes son la dirección IPv4.

Direcciones anycast

Las direcciones anycast de IPv6 se asignan para más de un interface de red, de tal modo que, un paquete enviado con este direccionamiento, es recibido por el interface más cercano, hablando en términos de la métrica de IP.

Las nuevas extensiones de cabecera logran un mayor rendimiento del tráfico al pasar por los nodos intermedios

Uno de los efectos resaltar del uso de direcciones anycast, es que permiten lo que se denomina "source selected policies", lo que habilita a los nodos para "elegir" entre varios proveedores de servicios, aquel que desean que cargue con su tráfico. Esto se consigue configurando varias direcciones anycast para identificar un conjunto de routers; una dirección para cada uno de los proveedores del conjunto). Otra posibilidad es utilizar este tipo de direcciones para identificar un grupo de routers de una subnet o que dan entrada a un determinado dominio.

Las direcciones anycast se ubican en el espacio de direcciones unicast y, por tanto, son sintácticamente indistinguibles de éstas. Dicho de otra forma, cuando una dirección unicast es asignada a más de un interface, nos encontramos ante una dirección anycast, y los nodos afectados deben configurarse explicitamente para saber que forman parte de un entorno anycast.

Direcciones multicast

Una dirección multicast es un identificador único para un grupo de interfaces. Además, cada interface puede pertenecer a cualquier número de grupos multicast.

Las direcciones multicast tienen el siguiente formato:

- **Prefix:** los ocho primeros bits de una dirección IPv6 a uno, la identifican como una dirección multicast.

Las nuevas direcciones IPv6 se dividen en tres grandes grupos: direcciones Unicast, direcciones Anycast y direcciones Multicast

- **Flags:** los cuatro bits siguientes forman un conjunto de cuatro flags. Los tres primeros están reservados y deben ser inicializados a cero. El último significa, si es 0, que se trata de una dirección multicast permanentemente asignada (Well-known Multicast Address) por el IANA. Si es 1, se trata de una dirección no permanente.

- **Scope**: Son también cuatro bits, y se utilizan para limitar el alcance de un grupo multicast. Los posibles valores son los siguientes:

0 (Reservado).
1 Ambito de nodo local.
2 Ambito de enlace local.
3 (No asignado).
4 (No asignado).
5 Ambito de "site" local.
6 (No asignado).
7 (No asignado).
8 Ambito de organización local.
9 (No asignado).
A (No asignado).
B (No asignado).
C (No asignado).
D (No asignado).
E Ambito global.
F (Reservado).

Group ID: Los 112 bits que quedan identifican el grupo multicast según esté definido, como permanente o temporal, en el campo Scope.

El routing con IPng

IPng incluye algunas extensiones simples que, sin embargo, le dotan de nuevas y poderosas funcionalidades, como son:

- Capacidad de selección de rutas basada en distintas políticas, costes, etc.
- Movilidad de hosts.
- Reasignación automática de direcciones.

El routing con IPng permite la reasignación automática de direcciones

Todo esto se consigue mediante la inclusión de listas de uno o más nodos intermedios a ser visitados, así como de un registro del camino de vuelta. Esta función es similar a las opciones "Loose Source" y "Record Route" existentes en IPv4.

Mejoras en la calidad de servicio

Los campos "Flow Label" y "Priority" de la cabecera de IPv6 permiten la identificación de paquetes que requieren tratamientos especiales por parte de los routers. En esto consisten las mejoras anunciadas de "soporte a aplicaciones en tiempo real" y "soporte a aplicaciones multimedia". El uso de estas características sigue en fase experimental y, por lo tanto, sujeto a cambios.

Flow Labels es un campo de 24 bits generado a ceros por los routers o hosts que no soporten IPv6, respetado si no son ellos quienes generan el paquete, o ignorado si son los receptores.

Un "flow" es una secuencia de paquetes enviados desde un determinado origen a un determinado destino, unicast o multicast, para los cuáles, y a instancias del emisor, se requiere un manejo especial por parte de los routers intermediarios. La naturaleza de ese manejo especial se determina, bien mediante un protocolo de control, tal como un protocolo de reserva de recursos, bien por la información contenida en los propios paquetes, tal como opciones hop-by-hop.

Todos los paquetes pertenecientes al mismo flujo, deben ser enviados desde la misma dirección origen, hacia la misma dirección destino, y con el mismo valor, distinto de cero, en el campo flow label.

El campo "Priority", de 4 bits, habilita a un origen para identificar la prioridad con que desea que determinados paquetes sean manejados, en relación con otros paquetes del mismo origen. Los valores para Priority están divididos en dos rangos:

- Los valores desde 0 hasta 7 se usan para especificar la prioridad de tráfico sujeto a congestión.
- Los valores desde 8 hasta 15, se usan para tráfico independiente de la congestión, como por ejemplo, los paquetes de aplicaciones en tiempo real.

La privacidad y los mecanismos de autentificación se solucionan mediante dos opciones integradas para servicios de seguridad: IPng Authentication Header y IPng Encapsulating Security Header

Para tráfico sujeto a congestión, se recomiendan los siguientes valores según categorías:

0. Tráfico no tipificado.
1. "Filler" traffic. (News, etc).
2. Tranferencias que no requieren atención. (Mail, etc).
3. Reserved.
4. Tráfico atendido. (FTP, HTTP, NFS, etc).
5. Reserved.
6. Tráfico interactivo. (Telnet, X, etc).
7. Tráfico de control Internet. (Protocolos de routing, SNMP, etc).

Para tráfico no sujeto a congestión, el valor más bajo de prioridad, (8), se usa

para paquetes que el emisor considera más susceptibles de ser descartados bajo condiciones de congestión, y el más alto, (15), para los menos susceptibles. Por ejemplo, se podrían considerar más "descartables" los paquetes de vídeo que los de audio.

La seguridad en IPng

La Internet actual tiene determinados problemas de seguridad, en cuanto a la privacidad y a los mecanismos de autentificación, que IPng pretende solucionar. La solución se presenta mediante dos opciones integrada‡ s para servicios de seguridad. Estas opciones pueden ser utilizadas conjuntamente o por separado para proveer de diferentes niveles de seguridad a diferentes usuarios.

El primer mecanismo, llamado IPng Authentication Header, es una extensión de la cabecera que confiere autentificación e integridad (sin confidencialidad) a los datagramas IPng. Puede soportar diversas técnicas de autentificación.

El segundo, llamado IPng Encapsulating Security Header, es similar a los protocolos de seguridad, (SP3D, ISO NLSP) pero más flexible e independiente del algoritmo.

Los mecanismos de transición hacia IPng

El objetivo clave de la transición es el de permitir la interoperación entre las máquinas que funcionan con IPv4 y las que lo hacen con IPv6. Otro objetivo es el de permitir a los hosts y routers IPv6 desplegarse por la Internet de una manera incremental y con pocas interdependencias. Un tercer objetivo es que la transición sea tan fácil como sea posible para los usuarios finales, los administradores de sistemas y los operadores de redes.

Los mecanismos de transición hacia IPng contienen cierto número de características, incluyendo las siguientes:

1. Desarrollo y actualización incremental: cada host o router IPv4 individual puede ser actualizado a IPv6 sin requerir que otros hosts o router sean actualizados al mismo tiempo.

2. Dependencias de actualización mínimas: el único prerrequisito para actualizar los hosts a IPv6 es que su servidor de DNS sea actualizado para poder tratar registros de direcciones IPv6. (No hay prerrequisitos a la hora de actualizar routers).

3. Facilidad de direccionamiento: cuando son actualizados hosts o routers ya existentes, pueden continuar utilizando sus antiguas direcciones IPv4, es decir, no necesitan la asignación de nuevas direcciones y los administradores no necesitan trazar nuevos planes de direccionamiento.

4. Bajo coste de puesta en marcha: ningún o casi ningún trabajo se necesita para llevar a cabo la actualización a IPv6.

El objetivo clave de la transición es el de permitir la interoperación entre las máquinas que funcionan con IPv4 y las que lo hacen con IPv6

Estas características son posibles gracias a que los mecanismos empleados son los siguientes:

1. Una estructura de direccionamiento IPv6 donde quedan embebidas las direcciones IPv4, codificando la información utilizada por los mecanismos de transición.

2. Un modelo de desarrollo donde todos los hosts y routers actualizados a IPv6 son capaces, durante la fase de transición, de comportarse dualmente como máquinas IPv4 o como máquinas IPv6.

3. La técnica de encapsulamiento de IPv6 sobre IPv4 para ser transportados sobre routers que todavia no han sido actualizados.

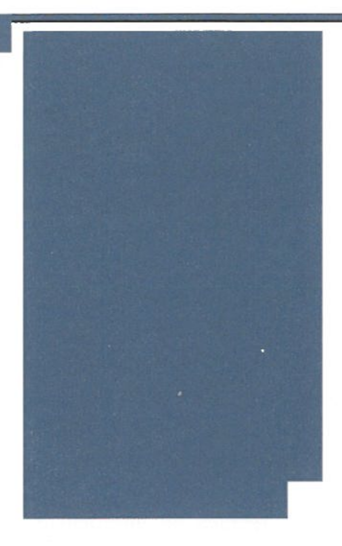
4. La técnica de traslación de cabeceras para permitir la eventual introducción de topologías de routing que sólo soporten tráfico IPv6. El uso de esta técnica es opcional, y será usado en posteriores fases de la transición, si al final se decide usarla.

Conclusión

La próxima versión de IP no es sólo un modelo preparado cuidadosamente para cubrir todas las necesidades que el uso de IPv4 ha ido haciendo patentes, sino que ha demostrado ser, con la experiencia vivida hasta el momento, una actualización de protocolo robusta y fiable.

Es encomiable el esfuerzo que se ha invertido en su desarrollo, y que ha hecho de IPng un protocolo de transición extremadamente sencilla, a pesar de lo revolucionario de sus innovaciones.

Estamos seguros de que en los próximos años veremos la implantación y evolución de la nueva versión, la comentada en este artículo, con la certeza de que IPng será uno de los pilares de la Internet futura, algunas de cuyas prestaciones y capacidades aún no alcanzamos a imaginar.



Correo del lector

En esta sección, los lectores de SÓLO PROGRAMADORES tienen la oportunidad de hallar respuesta a los problemas que puedan tener en cualquier tema relacionado con la programación.

Pregunta

Hola, soy un fiel lector de vuestra revista desde hace ya más de un año, mi nombre es Carlos Arroba y soy un fanático de la informática, especialmente de los lenguajes de programación y... de los juegos de ordenador.

Una de mis grandes cadencias profesionales ha sido todo lo relacionado con el sistema operativo Unix, al cual no tengo acceso desde mi PC.

Hace tiempo, un amigo me comentó que existe una versión "gratis" de Unix, que creo se llama Linux y me dijo que vosotros lo habías incluido en un especial. He comprado la revista de este mes y, efectivamente, he visto una página de publicidad que hace referencia al especial Linux y mi pregunta es ¿Cuándo lo pensáis sacar? ¿Se venderá por separado o será una revista mensual? ¿Pensáis sacar más especiales?

Tengo una segunda pregunta, estoy empezando a programar en Internet pero tengo problemas en realizar páginas que se vean bien con todos los navegadores. ¿Qué puedo hacer? ¿Hay algún código estándar que se pueda utilizar? He leido que existe un organismo que se encarga de normalizar el HTML, me podríáis dar su dirección Internet.

Muchas gracias.

Figura 1.



Respuesta

Estimado Carlos, hemos recibido un gran número de cartas, sobre todo a través de nuestra dirección de correo solop@towercom.es, de lectores que han tenido el mismo problema que tú, y hemos seleccionado la tuya porque creemos, que con ella se solucionarán un gran número de dudas con referencia a este número especial de Linux..

El especial Linux, en el que se incluía la versión Linux Debian 1.3.1 en dos discos CD-ROM, ya salió a la venta, lo que ha pasado es que tuvo tanta aceptación que se agotó enseguida.

Si eres uno de los lectores, que se quedaron sin poder comprarlo, puedes solicitarlo por carta, fax o e-mail (sucrip@towercom.es) a través de nuestro departamento de suscripciones y números atrasados (todos estos datos los podrás encontrar en el staff de la revista, que hay en las primeras páginas junto con el editorial, que se publica cada mes en la revista).

El número especial de Linux es un número independiente de los otros doce números, que sacamos al año y no es el único que ha salido.

Hace cuatro meses sacamos un segundo especial, dedicado a la programación en Internet y tenemos previsto realizar números especiales cada vez que consideremos que exista algún tema de actualidad, que tenga el suficiente interés para vosotros, como podría ser Java, Redes, etc.

Os recordamos que la mayoría de colecciones temáticas y contenidos de la revista los decidís vosotros cuando nos explicáis cuáles son vuestras inquietudes y sugerencias. Aunque podéis enviarnos faxes y cartas, os rogamos que si os es posible, nos enviéis correos electrónicos por Internet, ya que la comunicación será más fluida. En cuanto a tu segunda pregunta, te recomendamos que visites la siguiente dirección (donde encontrarás respuestas a tus preguntas):
http://www.w3c.com

Slackware v 3.4, Database Designer Demo v 1.0, Folkweb Server v1.0

CONTENIDO DEL CD

En este número, el equipo de Sólo Programadores ha decidido dar una grata sorpresa a todos los forofos de Linux. Y es que este mes hemos incluido la recien estrenada distribución **Slackware 3.4**. Los 'Linuxeros' que utilicen esta distribución o quieran instalarla verán que se ha actualizado el Kernel de la misma a la revisión 2.0.30 así como numerosos programas de los paquetes de aplicaciones como

- PPP daemon 2.2.0f
- Dynamic linker (ld.so) 1.9.5
- GNU CC 2.7.2.3
- Binutils 2.8.1.0.1
- Linux C Library 5.4.33
- Linux C++ Library 2.7.2.8

Folkweb.

- Termcap 2.0.8
- Procps 1.2.2
- Gpm 1.10
- SysVinit 2.69
- Util-linux 2.7.1

En el apartado de redes y a solicitud de nuestros lectores podréis encontrar un servidor de páginas Web de lo más completo llamado **FolkWeb Server** para Windows 95 / NT, y la versión de demostración sin límite de tiempo del software de Oracle "Database Designer". Esta demo permite crear hasta 10 tablas.

En el apartado de la programación en Internet, y como sabemos que los tutoriales son siempre bien recibidos, aquí tenéis uno sobre la programación en VRML bastante completito. Se encuentra en formato de páginas web navegables.

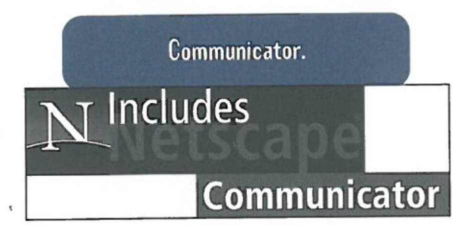
Incluida también en esta sección está una gran cantidad de información acerca del proyecto Eurométodo. Eurométodo es una metodología europea para la adquisición de sistemas de información y servicios relacionados. **EUROMÉTODO version 1** consta de una Guía de Referencia, Diccionario y Anexos. Esta versión ha sido publicada en Julio de 1996 e inicialmente sólo está disponible en inglés, aunque la versión aquí introducida es una traducción al español de la inglesa.

Se trata de una metodología de carácter público, promovida por la Comisión Europea cuya documentación está disponible en formato PDF. Para visualizarlo es necesario disponer de un visualizador PDF como Adobe Acrobat Reader.

Descripción General

Eurométodo es un proyecto de la Comisión Europea (DG III, Industry, Legislation and Standardization and Telematics net-

Communicator.



works) auspiciado por el PPG, Grupo de Compras Públicas en el que está representadas las Administraciones de los Estados miembros y que también desarrolla EP-HOS. Bajo contrato con la Comisión han intervenido en el desarrollo de Eurométodo algunas de las más importantes compañías de software y servicios europeas agrupadas en el consorcio *Eurogroup*, liderado por Sema-Group Francia.

El resultado, Eurométodo v1, es un documento con dos orientaciones diferentes: como marco metodológico y como método propiamente dicho.

Como marco metodológico proporciona un conjunto de conceptos y una terminología:

- para mejorar la relación cliente-proveedor
- para armonizar los métodos

Como método sirve para definir, planificar y ejecutar la adquisición de un sistema de información y los servicios relacionados. Se utiliza para valorar y determinar:

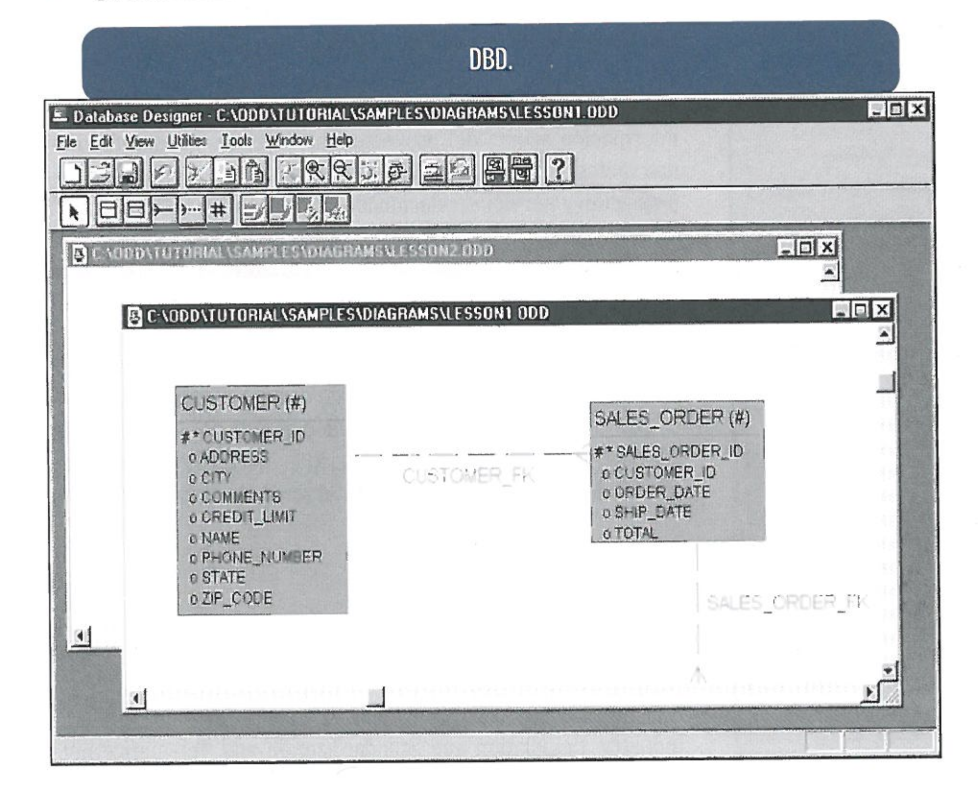
- la situación del problema y los riesgos asociados
- el objetivo de la adquisición
- la estategia de la adquisición, de la adaptación del sistema de información y de la provisión de servicios
- el "plan de entregas", el aspecto más relevante de las relaciones cliente-proveedor a nivel contractual.

Eurométodo no se ocupa de aspectos legales ni es un método de desarrollo de software.

Eurométodo en España

El Ministerio de Administraciones Públicas ha participado en la iniciativa Eurométodo desde su lanzamiento, incluido un ejercicio de validación de Eurométodo version 1 en base a un proyecto propio. Por otra parte, funciona desde marzo de 1994 el Foro Español de Eurométodo (FOREM) en el que se dan cita todas las instancias interesadas en esta iniciativa: Administraciones públicas, proveedores informáticos, asociaciones de profesionales y organizaciones de normalización y calidad.

DBD.



Resumen CD-ROM.

- Slackware v 3.4
- Database Designer Demo v 1.0
- Folkweb Server v1.0
- Información Eurométodo v 1
- Tutorial VRML
- WWW Gif Animator v 1.1
- Y mucho más…

Varios

Siguiendo la política empezada hace unos cuantos números, vamos a ir proporcionando las constantes actualizaciones del índice de aplicaciones disponibles para Linux denominado Linux Software Map. En este número podéis encontrar su última revisión así como las nuevas versiones de los ficheros de datos del software ViruScan de McAfee.

Por último destacar un programa de animación de Gifs llamado **WWW Gif Animator v1.1** ideal para los diseñadores de páginas Web que pueden hacer uso de él si los requerimientos de estética no son tan elevados como para precisar de la programación Java.

Despedida

Bueno, como cada mes nos despedimos, ya sabéis dónde nos podéis encontrar para enviarnos cualquier consulta, petición, consejo, queja (esperemos que no), etc. Nos vemos el mes que viene con muchas más aplicaciones y utilidades. Chao!



El nuevo gestor DB2 Universal Database, trabaja en gran cantidad de plataformas, empezando por Windows NT.

¿Le falta algo a su base de datos?

La nueva versión de DB2 incorpora Java como lenguaje nativo, y es capaz de combinar en su red información en imágenes, audio y video, junto con sus datos tradicionales. Le ofrece las ventajas de disponer de una tecnología de base de datos que se refleja en la integración Internet e intranet más completa, así como el ser una solución abierta, segura y escalable para sus datos convencionales y multimedia. Adivine todo lo que se está perdiendo. Pruebe el nuevo DB2 Universal Database. Si desea más información sobre el producto y como obtener un CD de demostración, llame al 900 100 400, de lunes a viernes de 9 a 19 horas o visítenos en: www.software.ibm.com/info/db2, o en www.ibm.es



Soluciones para nuestro pequeño mundo